Internet Payment Gateway

Integrationsleitfaden

API

First Data Austria GmbH, Leonard-Bernstein-Straße 10, 1220 Wien
© 2015 | First Data Austria GmbH. Alle Rechte vorbehalten.

Internet Payment Gateway

INTEGRATIONSLEITFADEN
API

Version 4

Inhalt

1 Einführung 7
2 Notwendige Elemente 8
3 So funktioniert die API 9
4 Transaktionen an das Gateway senden 11
5 Erstellung von Transaktionen in XML 13
5.1 Kreditkarten / Lastschrift-Transaktionen 13
5.1.1 Sale (Verkauf) 13
5.1.2 PreAuth (Vorautorisierung) 14
5.1.3 PostAuth (Buchung einer Vorautorisierung) 14
5.1.4 ForcedTicket (Buchung telefonisch erfolgter Vorautorisierungen) 15
5.1.5 Return (Rückzahlung) 15
5.1.6 Credit (Gutschrift) 16
5.1.7 Void (Storno) 16
5.2 ClickandBuy 18
5.2.1 Wiederkehrende Bezahltransaktionen 18
5.2.2 Return (Rückzahlung) 18
5.2.3 Void (Storno) 19
5.2.4 Credit (Gutschrift) 19
5.2.5 Transaktionsstatus 20
5.3 PayPal 21
5.3.1 Wiederkehrende Zahlungen 21
5.3.2 Return 22
5.3.3 Stornierung 23
Das aufgeführte XML Statement ist ein Beispiel für eine Stornierungs Transaktion mit dem minimalen Set an Befehlen: 23
5.3.4 Gutschrift 23
5.4 SOFORT Überweisung 25
5.4.1 Return 25
6 Zusätzliche Aktionen über den Web-Service 25

6.1 Manueller Kassenschnitt 25
6.2 Anfragen zu einer Bestellung (Inquiry Order) 26
6.3 Get Last Orders 30
6.3.1 Latest orders of a Store 30
6.3.2 Latest orders of a Store within a given date range 30
6.3.3 Alle Bestellungen eines Shops für eine bestimmte OrderID 31
Dieser Befehl unterstützt eine umfangreiche Abfrage von durchgeführten Bestellungen eines bestimmten Shops in einem definierten Zeitraum 31
6.3.4 Response 31
6.4 Get Last Transactions 36
6.4.1 Die letzten Transaktionen eines Shops 37
6.4.2 Alle Transaktionen eines Shops und einer definierten TransactionID 37
6.4.3 Response 38
6.5 Wiederkehrende Zahlungen 40
6.6 Validierung von Kreditkartendaten 42
6.7 Transaktionsstatus 45
Einige Payment Endpoints senden das Transaktionsergebnis nicht in Ihrer Antwort mit. In diesen Fällen sendet das IPG einen Bestätigungscode der mit einem Fragzeichen beginnt (?:..) 45
6.8 Anfrage von email notifications 45
6.9 Warenkorbinformationen und Produktkatalog 45
6.9.1 Warenkorbinformationen in Transaktionsbenachrichtigungen 45
6.9.2 Einen Produktkatalog aufsetzen 46
6.9.3 Den Lagerbestand managen 48
6.9.4 Sale Transaktionen und Lagerbestand 49
7 Erweiterungsfunktion Data Storage 51
7.1 Speicherung/Aktualisierung von Zahlungsinformationen bei einer Transaktion 51
7.2 Zahlungstransaktionen mit hinterlegten Daten 51
7.3 Hinterlegen eines Datensatzes ohne gleichzeitige Zahlungstransaktion 52
7.4 Sperrung der Verwendung von gleichen Zahlungsinformationen für mehrere Datensätze 53
7.5 Abruf hinterlegter Datensätze 53
7.6 Löschen hinterlegter Datensätze 54
7.7 Prüfen bestehender Datensatze 54
7.8 Löschen bestehender Datensatze 55
8 Erweiterungsfunktion Pay&Call 56
9 Übersicht der XML-Tags 57
9.1 Übersicht nach Transaktionstyp 57
9.2 Beschreibungen der XML-Tags 62

9.2.1 CreditCardTxType 62
9.2.2 CreditCardData 63
9.2.3 CreditCard3DSecure 63
9.2.4 ClickandBuyTxType 64
9.2.5 ClickandBuyData 64
9.2.6 PayPalTxType 64
9.2.7 Payment 64
9.2.8 TransactionDetails 65
9.2.9 Billing 66
9.2.10 Shipping 67
9.2.11 TopUpTxType 67
9.2.12 ClientLocale 68
10 SOAP Anfragenachricht erstellen 69
10.1 Beschreibung der XML-Tags 70
10.1.1 CreditCardTxType 70
10.1.2 CreditCardData 70
10.1.3 CreditCard3DSecure 71
10.1.4 ClickandBuyTxType 71
10.1.5 ClickandBuyData 71
10.1.6 PayPalTxType 72
10.1.7 Payment 72
10.1.8 TransactionDetails 72
10.1.9 InquiryRateReference 73
10.1.10 Billing 74
10.1.11 Shipping 74
10.1.12 TopUpTxType 75
10.1.13 ClientLocale 75
11 SOAP-Antwortnachricht lesen 77
11.1 SOAP-Antwortnachricht 77
11.2 SOAP-Fehlernachricht 78
11.2.1 SOAP-ENV:Server 78
11.2.2 SOAP-ENV:Client 79
12 Transaktionsergebnis auswerten 82
12.1 Erfolgreiche Transaktionen 82
12.2 Fehlerhafte Transaktionen 83
13 HTTPS POST-Anfrage erstellen 86
13.1 PHP 87

13.1.1 cURL PHP-Erweiterung nutzen 87
13.1.2 cURL-Befehlszeilentool nutzen 87
13.2 ASP 88
14 SSL-Verbindung aufbauen 89
14.1 PHP 89
14.1.1 PHP-cURL-Erweiterung nutzen 89
14.1.2 cURL-Befehlszeilentool nutzen 90
14.2 ASP 90
15 HTTPS POST-Anfrage senden und die Antwort empfangen 93
15.1 PHP 93
15.1.1 PHP-cURL-Erweiterung nutzen 93
15.1.2 cURL-Befehlszeilentool nutzen 93
15.2 ASP 94
16 Verwendung eines Java Client zur Nutzung des Web Service 95
16.1 Erzeugung einer IPGApiClient Instanz 95
16.2 Transaktionen erstellen und Antwort verarbeiten 95
16.3 Erstellen einer Aktion 96
16.4 Einsatz eines Proxys 96
17 Anhang 97
ipgapi.xsd 97
v1.xsd 101
a1.xsd 130
Fehleranalyse - Merchant Exceptions 145
Fehleranalyse - Processing Exceptions 150
Fehleranalyse - Login-Fehler bei der Verwendung von cURL 155
Fehleranalyse - Login-Fehler bei der Verwendung des Java Client 157

Unterstützung bei Fragen

Für das First Data Austria Internet Payment Gateway sind verschiedene Handbücher erhältlich. Der vorliegende API Integrationsleitfaden ist für die technische Integration besonders nützlich. Daneben finden Sie im Benutzerhandbuch First Data Austria Virtual Terminal

wichtige Hinweise zum Abrufen von Berichten und Grafiken zu Ihren Zahlungsvorgängen sowie zur Transaktionsverarbeitung durch manuelle Eingabe der Daten.

Wenn Sie in der verfügbaren Dokumentation keine Antwort auf Ihre Fragen finden, hilft Ihnen gerne unser Support Team weiter:

Tel: +43 800 005 680. Der Anruf ist für Sie kostenlos.

E-Mail: Technik_FirstData_Austria@telecash.de

1 Einführung

Die First Data Austria API ist eine Schnittstelle für die Anwendungsprogrammierung (API, Application Programming Interface). Sie ermöglicht Ihnen die Anbindung einer eigenen Applikation an das First Data Austria Internet Payment Gateway. Auf diese Weise kann Ihre Applikation Zahlungen ohne jeglichen Anwendereingriff einreichen.

Bitte beachten Sie, dass bei einer Speicherung oder Verarbeitung von Kreditkarten-Inhaberdaten in Ihrer eigenen Applikation unbedingt die Sicherheitsstandards der Kreditkartenindustrie (PCI DSS) einzuhalten sind und je nach Transaktionsvolumen eine Prüfung der von Ihnen verwendeten Systemkomponenten durch einen qualifizierten Sicherheitsprüfer notwendig sein kann.

Technisch gesehen ist First Data Austria API ein Web-Service, der an einem entfernten Standort die Abwicklung von Transaktionen übernimmt. Diese Auslegung bietet drei wesentliche Vorteile, die nachfolgend zusammenfassend dargelegt sind:

- **Plattform-Unabhängigkeit:** Kommunikation mit dem First Data Austria API Web Service bedeutet, dass Ihre Applikation nur in der Lage sein muss, SOAP-Nachrichten zu senden und zu empfangen. Es gibt hierbei keine mit einer spezifischen Plattform verbundenen Anforderungen, da der Web Service Standard auf einem Bestand an offenen Standards basiert. Kurz gesagt sind Sie frei in der Wahl einer beliebigen Programmiersprache (z. B. J2EE, .NET, PHP, ASP usw.), die Ihre Applikation in die Lage versetzt, mit dem First Data Austria API Web Service zu kommunizieren.

- **Einfache Integration:** Die Kommunikation mit einem Web Service ist einfach. Ihre Applikation muss dazu eine SOAP-Anfragenachricht erstellen, die Ihre Transaktion enthält, sie via HTTPS an den Web Service senden und auf eine SOAP-Antwortnachricht mit dem Statusbericht zu Ihrer Transaktion warten. Da SOAP und HTTP mit dem Ziel einer gewissen Übersichtlichkeit entwickelte Protokolle sind, ist die Erstellung von Anfragen und Antworten nicht sehr schwierig. Außerdem müssen Sie dies nur selten selbst programmieren, da es in nahezu jeder Programmiersprache hierfür ein reichhaltiges Angebot an Bibliotheken gibt. Im Allgemeinen benötigt man für die Erstellung einer SOAP-Anfrage und die Verarbeitung der Antwort nur wenige ZeilenProgrammcode.

- **Sicherheit:** Sämtliche Kommunikation zwischen Ihrer Applikation und First Data Austria API ist SSL-verschlüsselt. Dies ermöglicht die Verwaltung eines Client-Zertifikats durch Ihre Applikation, das die Anwendung gegenüber dem Web Service eindeutig identifiziert. In gleicher Weise verwaltet First Data Austria API ein Server-Zertifikat, das Ihre Applikation dazu nutzen kann, die Identität des API Web Service zu überprüfen, ehe sie Daten dorthin sendet. Schließlich muss Ihre Applikation noch eine Basisautorisierung (mittels Benutzername / Passwort) vornehmen, ehe ihr die Kommunikation mit dem Web Service erlaubt wird. Sie stellt die Identität der Anwender sicher, die eine Autorisierung für die Kommunikation mit First Data Austria API besitzen. Diese beiden Sicherheitsmechanismen gewährleisten, dass die an First Data Austria API gesendeten Transaktionsdaten erstens vertraulich bleiben und zweites tatsächlich von Ihrer Applikation und von nirgendwo anders herstammen.

Dies ist nur eine kurze Zusammenfassung der von First Data Austria API gebotenen Leistungsmerkmale. Der Schwerpunkt dieses Leitfades liegt bei der Integration der First Data Austria API Funktionalität in Ihre Applikation. Wie Sie diese Integration erreichen, wird hier Schritt für Schritt erklärt.

2 Notwendige Elemente

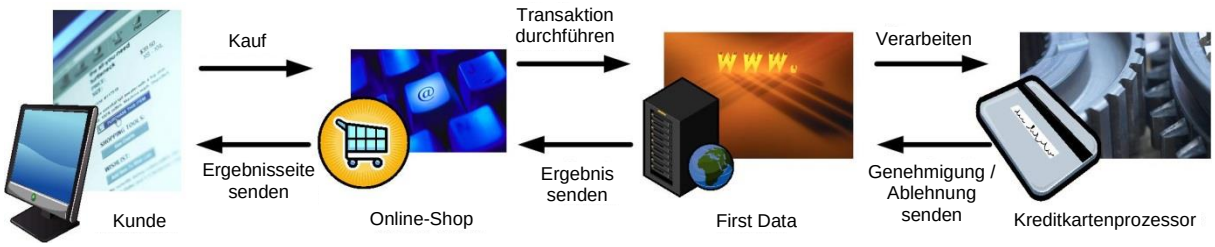
Die Aufrechterhaltung eines hohen Sicherheitsniveaus erfordert einige Elemente, die Sie für die sichere Kommunikation mit First Data Austria API benötigen. Da auf diese Elemente im weiteren Verlauf des Leitfadens immer wieder Bezug genommen wird, soll die folgende Checkliste Ihnen einen Überblick bieten. Anhand dieser Liste können Sie feststellen, ob Sie bereits alle benötigten Informationen erhalten haben:

- *Store ID:* Die Store-ID Ihres Onlineshops (z. B. `10012345678`). Sie wird für die Basisautorisierung benötigt.
- *User ID:* Die User-ID bezeichnet den Anwender, der das Zugriffsrecht auf First Data Austria API besitzt, z. B. `1`. Auch dieses Element wird für die Basisautorisierung benötigt.
- *Password:* Das für die Basisautorisierung benötigte Passwort.
- *Client Certificate p12 File:* Das in einer p12-Datei gespeicherte Client-Zertifikat. Der Dateiname folgt der Syntax `WS`*storeID*`._.`*userID*`.p12`. Im Beispielfall der oben genannten Store-ID / User-ID würde er `WS101._.1.p12` lauten. Diese Datei dient zur Authentifizierung des Clients gegenüber dem Internet Payment Gateway. Für eine Verbindung mit Java benötigen Sie ein *ks-File*, z. B. `WS10012345678._.1.ks`.
- *Client Certificate Installation Password:* Das für die Installation der p12-Datei mit dem Client-Zertifikat benötigte Passwort.
- *Client Certificate Private Key:* Der in einer Schlüssel-Datei gespeicherte private Schlüssel des Client-Zertifikats. Der Dateiname folgt der Syntax `WS`*storeID*`._.`*userID*`.key`. Im Beispielfall der oben genannten Store-ID / User-ID würde er `WS10012345678._.1.key` lauten. Einige Tools, die Sie bei der Einrichtung Ihrer Applikation für die Nutzung von First Data Austria API unterstützen, erfordern ein Passwort für die Client-Authentifizierung gegenüber dem Internet Payment Gateway.
- *Client Certificate Private Key Password:* Dieses Passwort schützt den privaten Schlüssel des Client-Zertifikats. Einige Tools, die Sie bei der Einrichtung Ihrer Applikation für die Nutzung von First Data Austria API unterstützen, erfordern dieses Passwort für die Client-Authentifizierung gegenüber dem Internet Payment Gateway. Seine Namenssyntax lautet: `ckp_`*creationTimestamp*. Ein Beispiel: `ckp_1193927132`.
- *Client Certificate PEM File:* Das in einer PEM-Datei gespeicherte Client-Zertifikat. Der Dateiname folgt der Syntax `WS`*storeID*`._.`*userID*`.pem`Im Beispielfall der oben genannten Store-ID / User-ID würde er `WS10012345678._.1.pem` lauten. Einige Tools, die Sie bei der Einrichtung Ihrer Applikation für die Nutzung von First Data Austria API unterstützen, erfordern anstelle der oben erwähnten p12-Datei diese Datei.
- *Server Certificate PEM File:* Das in der PEM-Datei `geotrust.pem` gespeicherte Server-Zertifikat. Es identifiziert den Server, auf dem First Data Austria API läuft. Für eine Verbindung mit Java benötigen Sie das truststore.ks-file.

3 So funktioniert die API

Im Folgenden wird die Funktionsweise von First Data Austria API anhand einer Kreditkarten-Transaktion erklärt. Andere Transaktionsarten erfolgen nach demselben Prinzip.

In den meisten Fällen leitet ein Kunde den Kommunikationsprozess im Ganzen ein, indem er Waren oder Dienstleistungen mit seiner Kreditkarte in Ihrem Online-Shop bezahlt. In der Folge sendet Ihr Shop eine Kreditkartentransaktion (in der Regel zur Abbuchung des Betrags vom Konto des Kunden) via First Data Austria API. Nach Empfang der Transaktion leitet das First Data Austria Internet Payment Gateway die Transaktion zur Genehmigung an den Kreditkartenprozessor weiter. Je nach Ergebnis wird die Genehmigung oder eine Fehlermeldung an Ihren Online-Shop zurückgesendet. Das bedeutet, dass sämtliche Aspekte der Kommunikation und Verarbeitung von der First Data Austria API Schnittstelle abgedeckt werden und Sie nur wissen müssen, wie Sie mit diesem Web Service kommunizieren.



Der Web Service Standard definiert eine solche Schnittstelle mittels Web Service Definition Language (WSDL). Im Falle von First Data Austria API finden Sie eine WSDL-Datei zur Definition des Web Service unter:

`https://test.ipg-online.com/ipgapi/services/order.wsdl`

Bitte beachten Sie, dass Sie Ihr Client-Zertifikat, Ihre Identifizierungsangaben und das Server-Zertifikat bereitstellen müssen, um die Datei in einem Web-Browser einzusehen oder abzurufen. Wenn Sie beispielsweise die WSDL-Datei in Microsoft Internet Explorer unter Microsoft Windows XP betrachten wollen, müssen Sie zuerst Ihr Client-Zertifikat und das Server-Zertifikat installieren und dann die oben genannte URL aufrufen. Dazu führen Sie folgende Schritte aus:

1. Öffnen Sie den Ordner, in dem Sie Ihre p12-Datei mit dem Client-Zertifikat abgelegt haben.
2. Doppelklicken Sie auf die p12-Datei mit dem Client-Zertifikat.
3. Klicken Sie auf *Weiter*. Prüfen Sie den Dateinamen (der bereits dem Pfadnamen der p12-Datei mit dem Client-Zertifikat entsprechen sollte) und klicken Sie auf *Weiter*.
4. Geben Sie das Passwort für die Installation des Client-Zertifikats ein und klicken Sie auf *Weiter*.
5. Wählen Sie die Option *Zertifikatspeicher automatisch auswählen* und klicken Sie auf *Weiter*. Damit platzieren Sie das Zertifikat in Ihrem persönlichen Zertifikatspeicher (genauer im persönlichen Zertifikatspeicher des lokal angemeldeten Windows-Anwenders).
6. Prüfen Sie die angezeigten Einstellungen und klicken Sie auf *Fertig stellen*. Damit ist Ihr Client-Zertifikat installiert.
7. Als nächstes müssen Sie das Server-Zertifikat installieren. Am einfachsten geschieht dies, indem Sie den Ordner öffnen, in dem Sie die PEM-Datei mit dem Server-Zertifikat gespeichert haben, und diese Datei in `geotrust.crt` umbenennen.
8. Doppelklicken Sie anschließend auf die umbenannte Datei mit dem Server-Zertifikat.

9. Klicken Sie auf *Zertifikat installieren*. Damit rufen Sie den gleichen Assistenten wie oben auf.
10. Klicken Sie auf *Weiter*. Wählen Sie *Alle Zertifikate in folgendem Speicher speichern* und suchen Sie den Ordner *Vertrauenswürdige Stammzertifizierungsstellen*. Klicken Sie auf *OK* und dann *Weiter*.
11. Prüfen Sie die angezeigten Einstellungen und klicken Sie auf *Fertig stellen* (eventuell müssen Sie die Installation nochmals bestätigen). Das Server-Zertifikat ist damit im Speicher des lokalen Computers für vertrauenswürdige Zertifikate installiert. Hier kann Microsoft Internet Explorer nach dem Server-Zertifikat suchen, um das empfangene First Data Austria API Server-Zertifikat zu überprüfen, wenn die oben genannte WSDL-URL aufgerufen wird.
12. Sie können jetzt ein Microsoft Internet Explorer Fenster öffnen und die oben genannte URL in das Adressenfeld eintragen.
13. Nach Aufruf der URL fordert der Server ihren Browser auf, das Client-Zertifikat zu senden, um sicherzustellen, dass er wirklich mit Ihrer Applikation kommuniziert. Da Sie zuvor das Zertifikat installiert haben, wird es ohne weitere Eingabeaufforderung an Sie an den Server gesendet (d. h. diesen Schritt bemerken Sie nicht). Anschließend sendet First Data Austria API das Server-Zertifikat (um sich gegenüber Ihrer Anwendung eindeutig auszuweisen). Dieses Zertifikat wird mit dem vertrauenswürdigen Zertifikat verglichen, das Sie zuvor installiert haben. Auch dies geschieht automatisch, so dass Sie keine Eingabe vornehmen müssen. Nun wird eine sichere Verbindung aufgebaut. Alle Daten, die zwischen Ihrer Applikation und dem First Data Austria API Web Service übertragen werden, sind somit SSL-verschlüsselt.
14. Als nächstes müssen Sie Ihre Identifizierungsangaben für die Autorisierung übermitteln. Als Benutzername müssen Sie Ihre Store-ID und User-ID in folgender Syntax eingeben: `WS`*storeID*`._.`*userID*. Beispiel: Wenn Ihre Store-ID `101` lautet, Ihre User-ID `007` und Ihr Passwort `myPW`, müssen Sie in das Feld User Name `WS101._.1` und in das Feld Password `myPW` eingeben. Ihre Identifizierungsangaben werden vor der Übermittlung an den Server verschlüsselt, da zu diesem Zeitpunkt bereits eine SSL-Verbindung besteht. Klicken Sie auf *OK*.
15. Die First Data Austria API WSDL-Datei wird angezeigt.

Die WSDL-Datei definiert die vom Web Service angebotenen Operationen, deren Eingabe- und Antwortparameter und wie diese Operationen aufgerufen werden können. Im Falle des First Data Austria API Web Service definiert sie nur eine Operation (`IPGApiOrder`), die durch Senden einer SOAP HTTPS-Anfrage an folgende URL aufrufbar ist:

```
https://test.ipg-online.com/ipgapi/services
```

Diese Operation verwendet eine XML-kodierte Transaktion als Input und liefert eine XML-kodierte Antwort zurück. Für die Nutzung von First Data Austria API ist es nicht notwendig ist, den Aufbau der WSDL-Datei zu verstehen. Die nachfolgenden Kapitel leiten Sie durch die Einrichtung Ihres Shops für die Erstellung und Einreichung von Kreditkartentransaktionen.

Wenn Sie Tools von Fremdanbietern zur Einrichtung Ihres Shops für den Zugriff auf First Data Austria API nutzen, müssen Sie eventuell die URL angeben, unter der die WSDL-Datei zu finden ist. Ähnlich wie oben beschrieben müssen Sie Ihrem Web Service Tool bekannt geben, dass die Kommunikation unter SSL-Verschlüsselung erfolgt und es darum erforderlich ist, dass Sie Ihr Client-Zertifikat senden und das Server-Zertifikat als vertrauenswürdig annehmen. Zudem müssen Sie Ihre Identifizierungsangaben senden. Wie dies geschieht, ist in erster Linie von Ihrem Web Service Tool abhängig. Lesen Sie die Einzelheiten in der Dokumentation des Tools nach.

4 Transaktionen an das Gateway senden

Dieses Kapitel soll Ihnen ein grundlegendes Verständnis von den Schritten vermitteln, die für die Einreichung von Transaktionen an das First Data Austria Internet Payment Gateway auszuführen sind. Es beschreibt, was geschieht, wenn ein Kunde mit seiner Kreditkarte bei einem Online-Shop bezahlt, das für die Einreichung von Transaktionen First Data Austria API einsetzt.

- Der Kunde klickt im Online-Shop auf die Schaltfläche *Bezahlen*.
- Der Online-Shop zeigt ein Formular an, das den Kunden zur Eingabe seiner Kreditkartennummer, Monat und Jahr des Ablaufdatums der Karte sowie der Kartenprüfnummer auffordert.
- Der Kunde nimmt die Eingaben vor und sendet die Daten an den Online-Shop ab (d. h. er kauft die Waren).
- Der Online-Shop empfängt die Daten und erstellt ein XML-Dokument für eine *Sale* Transaktion, das die vom Kunden gesendeten Daten sowie den vom Kunden zu zahlenden Gesamtbetrag enthält.
- Nach Erstellung der XML *Sale* Transaktion packt der Online-Shop diese in eine SOAP-Nachricht. Diese beschreibt die aufzurufende Web Service Operation und übermittelt die XML-Transaktion als Parameter dieser Operation.
- Nach Zusammenstellen der SOAP-Nachricht bereitet der Online-Shop diese für die Übertragung über das Internet vor, indem der Inhalt in eine HTTPS POST-Anfrage gepackt wird. Außerdem legt der Shop den HTTP-Header fest, insbesondere seine Identifizierungsangaben (diese müssen mit den Identifizierungsangaben übereinstimmen, die Sie für die Anzeige der WSDL Datei übermittelt haben).
- Danach baut der Online-Shop eine SSL-Verbindung auf, indem Client- und Server-Zertifikat übermittelt werden.
- Dann sendet der Online-Shop die HTTPS-Anfrage an den First Data Austria API Web Service und wartet auf eine HTTPS-Antwort.
- First Data Austria API nimmt die HTTPS Anfrage entgegen und liest die Autorisierungsinformationen aus, die der Shop in den HTTP-Header eingetragen hat.
- Nachdem erfolgreich überprüft wurde, dass der Shop zur Nutzung von First Data Austria API berechtigt ist, wird die im Body der HTTPS-Anfrage enthaltene SOAP-Nachricht ausgelesen und die Web Service Operation für die Handhabung der Transaktionsverarbeitung angestoßen.
- First Data Austria API führt die Transaktionsverarbeitung durch, erstellt ein XML-Antwortdokument, packt es in eine SOAP-Nachricht und sendet die SOAP-Nachricht im Body einer HTTPS-Antwort zurück an den Kunden.
- Der Eingang dieser HTTPS-Antwort veranlasst den Shop, die SOAP-Nachricht und das darin enthaltene XML-Antwortdokument auszulesen.
- In Abhängigkeit von den im XML-Antwortdokument enthaltenen Daten wird an den Kunden eine Genehmigungsseite gesendet, wenn die Transaktion erfolgreich war, und andernfalls eine Fehlerseite.
- Die Genehmigungs- oder Fehlerseite wird angezeigt.

Zusammengefasst muss Ihre Applikation folgende Schritte vollziehen, um Zahlungstransaktionen einzureichen und das Ergebnis auszuwerten:

- Ein XML-Dokument mit den kodierten Transaktionen erstellen
- Das XML-Dokument in eine SOAP-Anfragenachricht packen
- Eine HTTP POST-Anfrage erstellen, die im HTTP-Header die nötigen Informationen zur Identifikation Ihres Shops und im Body die SOAP-Anfragenachricht enthält
- Eine SSL-Verbindung zwischen Ihrer Applikation und First Data Austria API aufbauen

- Die HTTPS POST-Anfrage an das First Data Austria Internet Payment Gateway senden und die Antwort entgegennehmen
- Die SOAP-Antwortmitteilung aus dem Body der HTTPS-Antwort auslesen
- Das in der SOAP-Antwortmitteilung enthaltene XML-Antwortdokument analysieren

Diese sieben Schritte sind in den nachfolgenden Kapiteln beschrieben.

5 Erstellung von Transaktionen in XML

Dieses Kapitel beschreibt die Erstellung der verschiedenen Transaktionstypen in XML. Wie im vorangehenden Szenario ausgeführt, wird eine Transaktion erst in einem XML-Dokument kodiert, das seinerseits als Nutzlast in eine SOAP-Nachricht verpackt wird. Das bedeutet, die XML-kodierte Transaktion entspricht den Parametern, die an die First Data Austria API Web Service Operation übermittelt werden.

Bitte beachten Sie, dass es eine Vielzahl von Web Service Tools gibt, die Sie bei der Erstellung von Client-Stubs unterstützen und es ihnen vielleicht ersparen, sich mit dem eigentlichen XML-Format näher auseinanderzusetzen. Ein grundlegendes Verständnis des XML-Formats ist jedoch unabdingbar dafür, korrekte Transaktionen unabhängig von der verfügbaren Unterstützung durch Tools erstellen zu können. Darum sollten Sie sich mit dem XML-Format vertraut machen, das First Data Austria API für die Kodierung von Transaktionen verwendet.

5.1 Kreditkarten / Lastschrift-Transaktionen

Unabhängig vom Transaktionstyp sieht die grundlegende Struktur des XML-Dokuments einer Kreditkarten- / Lastschrift-Transaktion wie folgt aus:

```
<ipgapi:IPGApiOrderRequest
              xmlns:v1=http://ipg-online.com/ipgapi/schemas/v1
              xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
       <v1:Transaction>
              <v1:CreditCardTxType>...</v1:CreditCardTxType>
              <v1:CreditCardData>...</v1:CreditCardData>
              <v1:Payment>...</v1:Payment>
              <v1:TransactionDetails>...</v1:TransactionDetails>
              <v1:Billing>...</v1:Billing>
              <v1:Shipping>...</v1:Shipping>
       </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Das Element CreditCardTxType muss für alle Kreditkarten-Transaktionen angegeben werden. Die anderen Elemente sind abhängig vom Transaktionstyp. Der Transaktionsinhalt ist jedoch für jeden Transaktionstyp spezifisch.

XML Tags, die sich auf Card Present Transaktionen mit einem Chipkartenleser und PIN Eingabe beziehen, sie an dieser Stelle auf den Bereich XSD im Anhang des Dokuments verwiesen.

5.1.1 Sale (Verkauf)

Das folgende XML-Dokument entspricht einer *Sale* Transaktion mit dem Mindestbestand an Elementen:

```
<ipgapi:IPGApiOrderRequest
              xmlns:v1=http://ipg-online.com/ipgapi/schemas/v1
              xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
       <v1:Transaction>
              <v1:CreditCardTxType>
                     <v1:Type>sale</v1:Type>
              </v1:CreditCardTxType>
              <v1:CreditCardData>
                     <v1:CardNumber>4111111111111111</v1:CardNumber>
                     <v1:ExpMonth>12</v1:ExpMonth>
                     <v1:ExpYear>07</v1:ExpYear>
              </v1:CreditCardData>
```

```
            <v1:Payment>
                  <v1:ChargeTotal>19.00</v1:ChargeTotal>
                  <v1:Currency>978</v1:Currency>
            </v1:Payment>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

5.1.2 PreAuth (Vorautorisierung)

Das folgende XML-Dokument entspricht einer *PreAuth* Transaktion mit dem Mindestbestand an Elementen:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>preAuth</v1:Type>
            </v1:CreditCardTxType>
            <v1:CreditCardData>
                  <v1:CardNumber>4111111111111111</v1:CardNumber>
                  <v1:ExpMonth>12</v1:ExpMonth>
                  <v1:ExpYear>07</v1:ExpYear>
            </v1:CreditCardData>
            <v1:Payment>
                  <v1:ChargeTotal>100.00</v1:ChargeTotal>
                  <v1:Currency>978</v1:Currency>
            </v1:Payment>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

5.1.3 PostAuth (Buchung einer Vorautorisierung)

Das folgende XML-Dokument entspricht einer *PostAuth* Transaktion mit dem Mindestbestand an Elementen:

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>postAuth</v1:Type>
            </v1:CreditCardTxType>
            <v1:Payment>
                  <v1:ChargeTotal>59.00</v1:ChargeTotal>
                  <v1:Currency>978</v1:Currency>
            </v1:Payment>
            <v1:TransactionDetails>
                  <v1:OrderId>
                        703d2723-99b6-4559-8c6d-797488e8977
                  </v1:OrderId>
            </v1:TransactionDetails>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Im Falle, dass Ihr System nicht weiß, welche Paymentmethode für die Original-Vorautorisierung genutzt wurde, nutzen Sie PostAuth mit einem beliebigen TxType, welcher Post-Autorisierungen unterstützt. Das First Data Austria Internet Payment Gateway wählt die korrkte Paymentmethode an Hand der referenzierten OrderID.

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

5.1.4 ForcedTicket (Buchung telefonisch erfolgter Vorautorisierungen)

Das folgende XML-Dokument entspricht einer *ForceTicket* Transaktion mit dem Mindestbestand an Elementen:

```
<ipgapi:IPGApiOrderRequest
              xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
              xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
       <v1:Transaction>
              <v1:CreditCardTxType>
                     <v1:Type>forceTicket</v1:Type>
              </v1:CreditCardTxType>
              <v1:CreditCardData>
                     <v1:CardNumber>4111111111111111</v1:CardNumber>
                     <v1:ExpMonth>12</v1:ExpMonth>
                     <v1:ExpYear>07</v1:ExpYear>
              </v1:CreditCardData>
              <v1:Payment>
                     <v1:ChargeTotal>59.00</v1:ChargeTotal>
                     <v1:Currency>978</v1:Currency>
              </v1:Payment>
              <v1:TransactionDetails>
                     <v1:ReferenceNumber>123456</v1:ReferenceNumber>
              </v1:TransactionDetails>
       </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

5.1.5 Return (Rückzahlung)

Das folgende XML-Dokument entspricht einer *Return* Transaktion mit dem Mindestbestand an Elementen:

```
<ipgapi:IPGApiOrderRequest
              xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
              xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
       <v1:Transaction>
              <v1:CreditCardTxType>
                     <v1:Type>return</v1:Type>
              </v1:CreditCardTxType>
              <v1:Payment>
                     <v1:ChargeTotal>19.00</v1:ChargeTotal>
                     <v1:Currency>978</v1:Currency>
              </v1:Payment>
              <v1:TransactionDetails>
                     <v1:OrderId>
                            62e3b5df-2911-4e89-8356-1e49302b1807
                     </v1:OrderId>
              </v1:TransactionDetails>
       </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Im Falle, dass Ihr System nicht weiß, welche Paymentmethode für die Originaltransaktion genutzt wurde, nutzen Sie Return mit einem beliebigen TxType, welcher Returns unterstützt. Das First Data Austria Internet Payment Gateway wählt die korrkte Paymentmethode an Hand der referenzierten OrderID.

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

5.1.6 Credit (Gutschrift)

Bitte beachten Sie, dass für diese Funktion eine gesonderte Beauftragung und Freischaltung notwendig ist. Kontaktieren Sie hierzu bei Bedarf Ihren Ansprechpartner in unserem Vertriebsteam.

Das folgende XML-Dokument entspricht einer *Credit* Transaktion mit dem Mindestbestand an Elementen:

```
<ipgapi:IPGApiOrderRequest
              xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
              xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
       <v1:Transaction>
              <v1:CreditCardTxType>
                     <v1:Type>credit</v1:Type>
              </v1:CreditCardTxType>
              <v1:CreditCardData>
                     <v1:CardNumber>4111111111111111</v1:CardNumber>
                     <v1:ExpMonth>12</v1:ExpMonth>
                     <v1:ExpYear>07</v1:ExpYear>
              </v1:CreditCardData>
              <v1:Payment>
                     <v1:ChargeTotal>50.00</v1:ChargeTotal>
                     <v1:Currency>978</v1:Currency>
              </v1:Payment>
       </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

5.1.7 Void (Storno)

Das folgende XML-Dokument entspricht einer *Void* Transaktion mit dem Mindestbestand an Elementen:

```
<ipgapi:IPGApiOrderRequest
              xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
              xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
       <v1:Transaction>
              <v1:CreditCardTxType>
                     <v1:Type>void</v1:Type>
              </v1:CreditCardTxType>
              <v1:TransactionDetails>
                     <v1:OrderId>
                            62e3b5df-2911-4e89-8356-1e49302b1807
                     </v1:OrderId>
                     <v1:TDate>1190244932</v1:TDate>
              </v1:TransactionDetails>
       </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Im Falle, dass Ihr System nicht weiß, welche Paymentmethode für die Originaltransaktion genutzt wurde, nutzen Sie Void mit einem beliebigen TxType, welcher Voids unterstützt. Das First Data Austria Internet Payment Gateway wählt die korrkte Paymentmethode an Hand der referenzierten OrderID und TDate.

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

Im Falle, dass Ihr System nicht weiß, welche Paymentmethode für die Originaltransaktion genutzt wurde, nutzen Sie Void mit einem beliebigen TxType, welcher Voids unterstützt. Das First Data Austria Internet Payment Gateway wählt die korrkte Paymentmethode an Hand der referenzierten OrderID und TDate.

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

5.2 ClickandBuy

5.2.1 Wiederkehrende Bezahltransaktionen

Nach einer wiederkehrenden Bezahltransaktion für ClickandBuy, die über Connect oder das Virtuelle Terminal des First Data Austria Internet Payment Gateways eingereicht wurde, kann auch die API bis zur maximalen Transaktionszahl oder dem autorisierten Betrag genutzt werden.

Das folgende XML-Dokument entspricht einem Beispiel für eine PostAuth-Transaktion mit den minimal notwendigen Elementen.

```
<ns5:IPGApiOrderRequest
xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
xmlns:ns2="http://api.clickandbuy.com/webservices/pay_1_0_0/"
xmlns:ns3=http://ipg-online.com/ipgapi/schemas/a1 xmlns:ns4="http://ipg-
online.com/ipgapi/schemas/v1">
	<ns4:Transaction>
		<ns4:ClickandBuyTxType>
			<ns4:Type>postAuth</ns4:Type>
		</ns4:ClickandBuyTxType>
		<ns4:Payment>
			<ns4:ChargeTotal>1</ns4:ChargeTotal>
			<ns4:Currency>EUR</ns4:Currency>
		</ns4:Payment>
		<ns4:TransactionDetails>
			<ns4:OrderId>
	C-32121f4d-852f-4f48-8095-8585b917c079
	</ns4:OrderId>
		</ns4:TransactionDetails>
	</ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

5.2.2 Return (Rückzahlung)

Das folgende XML-Dokument entspricht einem Beispiel für eine Return-Transaktion mit den minimal notwendigen Elementen.

```
<ns5:IPGApiOrderRequest
xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
xmlns:ns2="http://api.clickandbuy.com/webservices/pay_1_0_0/"
xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1" mlns:ns4="http://ipg-
online.com/ipgapi/schemas/v1">
	<ns4:Transaction>
		<ns4:ClickandBuyTxType>
			<ns4:Type>return</ns4:Type>
		</ns4:ClickandBuyTxType>
		<ns4:Payment>
			<ns4:ChargeTotal>0.4</ns4:ChargeTotal>
			<ns4:Currency>EUR</ns4:Currency>
		</ns4:Payment>
		<ns4:TransactionDetails>
			<ns4:OrderId>
	C-32121f4d-852f-4f48-8095-8585b917c079
```

```
      </ns4:OrderId>
            </ns4:TransactionDetails>
      </ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

Im Falle, dass Ihr System nicht weiß, welche Paymentmethode für die Originaltransaktion genutzt wurde, nutzen Sie Return mit einem beliebigen TxType, welcher Returns unterstützt. Das First Data Austria Internet Payment Gateway wählt die korrkte Paymentmethode an Hand der referenzierten OrderID.

5.2.3 Void (Storno)

Das folgende XML-Dokument entspricht einem Beispiel für eine Void-Transaktion mit den minimal notwendigen Elementen.

```
<ns5:IPGApiOrderRequest
xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
xmlns:ns2="http://api.clickandbuy.com/webservices/pay_1_0_0/"
xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1" mlns:ns4="http://ipg-
online.com/ipgapi/schemas/v1">
      <ns4:Transaction>
            <ns4:ClickandBuyTxType>
                  <ns4:Type>void</ns4:Type>
            </ns4:ClickandBuyTxType>
            <ns4:TransactionDetails>
                  <ns4:OrderId>
      C-32121f4d-852f-4f48-8095-8585b917c079
      </ns4:OrderId>
                  <ns4:TDate>1321609898</ns4:TDate>
            </ns4:TransactionDetails>
      </ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

Im Falle, dass Ihr System nicht weiß, welche Paymentmethode für die Originaltransaktion genutzt wurde, nutzen Sie Void mit einem beliebigen TxType, welcher Voids unterstützt. Das First Data Austria Internet Payment Gateway wählt die korrkte Paymentmethode an Hand der referenzierten OrderID und TDate.

5.2.4 Credit (Gutschrift)

Bitte beachten Sie, dass für diese Funktion eine gesonderte Beauftragung und Freischaltung notwendig ist. Kontaktieren Sie hierzu bei Bedarf Ihren Ansprechpartner in unserem Vertriebsteam.

Das folgende XML-Dokument entspricht einer *Credit* Transaktion mit dem Mindestbestand an Elementen:

```
<ns5:IPGApiOrderRequest
xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:ns3="http://api.clickandbuy.com/webservices/pay_1_0_0/" mlns:ns4="http://ipg-
online.com/ipgapi/schemas/v1">
```

```
	<ns4:Transaction>
		<ns4:ClickandBuyTxType>
			<ns4:Type>credit</ns4:Type>
		</ns4:ClickandBuyTxType>
		<ns4:ClickandBuyData />
		<ns4:Payment>
			<ns4:ChargeTotal>1</ns4:ChargeTotal>
			<ns4:Currency>EUR</ns4:Currency>
		</ns4:Payment>
		<ns4:Billing>
			<ns4:Email>x@y.zz</ns4:Email>
		</ns4:Billing>
	</ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

Anders als andere Bezahlmethoden, beinhalten ClickandBuy-Transaktionen keine Kartennummer. Dafür benötigen diese Transaktionen die registrierte eMail-Adresse des Zahlungsempfängers. Die eMail-Adresse ist in das Feld ns4:Billing/ns:Email einzutragen.

5.2.5 Transaktionsstatus

Änderungen im Status von ClickandBuy-Transaktionen werden automatisch im ClickandBuy Merchent Messege System (MMS) angezeigt. Wenn Sie den letzten Status der Transaktion anfragen wollen, können Sie den API-Webservice GetExternalTransactionStatus nutzen.

```
<ns5:IPGApiActionRequest
xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
xmlns:ns2="http://api.clickandbuy.com/webservices/pay_1_0_0/"
xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1" xmlns:ns4="http://ipg-
online.com/ipgapi/schemas/v1">
	<ns3:Action>
		<ns3:GetExternalTransactionStatus>
			<ns3:OrderId>
	C-32121f4d-852f-4f48-8095-8585b917c079
	</ns3:OrderId>
			<ns3:TDate>1321603908</ns3:TDate>
		</ns3:GetExternalTransactionStatus>
	</ns3:Action>
</ns5:IPGApiActionRequest>
```

Nach einer erfolgreichen Transaktion ist der aktuelle Status in der Antwort im Feld TransactionState abrufbar.

```
<ipgapi:IPGApiActionResponse
xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi"
xmlns:a1="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:pay_1_0_0="ttp://api.clickandbuy.com/webservices/pay_1_0_0/"xmlns:v1=
"http://ipg-online.com/ipgapi/schemas/v1">
	<ipgapi:successfully>true</ipgapi:successfully>
	<a1:TransactionValues>
		<v1:Payment>
			<v1:ChargeTotal>1</v1:ChargeTotal>
			<v1:Currency>978</v1:Currency>
		</v1:Payment>
		<v1:TransactionDetails>
			<v1:OrderId>
	C-32121f4d-852f-4f48-8095-8585b917c079
	</v1:OrderId>
```

```
                <v1:TDate>1321603908</v1:TDate>
            </v1:TransactionDetails>
            <ipgapi:IPGApiOrderResponse>
                <ipgapi:ApprovalCode>
Y:000000:0009459525:PPX :1173994001
</ipgapi:ApprovalCode>
                <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                <ipgapi:OrderId>
C-32121f4d-852f-4f48-8095-8585b917c079
</ipgapi:OrderId>
                <ipgapi:PaymentType>CLICKANDBUY</ipgapi:PaymentType>
                <ipgapi:ProcessorApprovalCode>
000000
</ipgapi:ProcessorApprovalCode>
                <ipgapi:ProcessorReceiptNumber>
4001
</ipgapi:ProcessorReceiptNumber>
                <ipgapi:ProcessorCCVResponse />
                <ipgapi:ProcessorTraceNumber>
117399
</ipgapi:ProcessorTraceNumber>
                <ipgapi:TDate>1321603908</ipgapi:TDate>
                <ipgapi:TDateFormatted>
2011.11.18 09:11:48 (MEZ)
</ipgapi:TDateFormatted>
                <ipgapi:TerminalID>54000666</ipgapi:TerminalID>
            </ipgapi:IPGApiOrderResponse>
            <a1:ReceiptNumber>4001</a1:ReceiptNumber>
            <a1:TraceNumber>117399</a1:TraceNumber>
            <a1:TransactionState>CAPTURED</a1:TransactionState>
            <a1:SubmissionComponent>API</a1:SubmissionComponent>
        </a1:TransactionValues>
</ipgapi:IPGApiActionResponse>
```

5.3 PayPal

5.3.1 Wiederkehrende Zahlungen

Nachdem eine wiederkehrende Zahlungsautorisierung for PayPal über die Connect Schnittstelle ausgelöst wurde, kann für weitere wiederkehrende Zahlungen die Web Service API Schnittstelle verwendet werden,

Das folgende XML Statement ist ein Beispiel für eine PostAuth Transaktion mit dem minimalen Set an Befehlen:

```
<ns5:IPGApiOrderRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/v1">
        <ns4:Transaction>
            <ns4:PayPalTxType>
                <ns4:Type>postAuth</ns4:Type>
            </ns4:PayPalTxType>
            <ns4:Payment>
                <ns4:ChargeTotal>1</ns4:ChargeTotal>
```

```
            <ns4:Currency>EUR</ns4:Currency>
        </ns4:Payment>
        <ns4:TransactionDetails>
            <ns4:OrderId>
                C-32121f4d-852f-4f48-8095-8585b917c079
            </ns4:OrderId>
        </ns4:TransactionDetails>
    </ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

Eine detaillierte Beschreibung aller Elemente erhalten Sie in dem Kapitel Übersicht der XML Tags.

5.3.2 Return

Das aufgeführte XML Statement ist ein Beispiel für eine Return Transaktion mit dem minimalen Set an Befehlen:

```
<ns5:IPGApiOrderRequest
        xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
        xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
        mlns:ns4="http://ipg-online.com/ipgapi/schemas/v1">
    <ns4:Transaction>
        <ns4:PayPalTxType>
            <ns4:Type>return</ns4:Type>
        </ns4:PayPalTxType>
        <ns4:Payment>
            <ns4:ChargeTotal>0.4</ns4:ChargeTotal>
            <ns4:Currency>EUR</ns4:Currency>
        </ns4:Payment>
        <ns4:TransactionDetails>
            <ns4:OrderId>
                C-32121f4d-852f-4f48-8095-8585b917c079
            </ns4:OrderId>
        </ns4:TransactionDetails>
    </ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

In case your system is not aware of the payment method that has been used for the original transaction, the Return can be performed using any TxType which supports Returns. The gateway will then select the correct payment method based on the referenced Order ID.

5.3.3 Stornierung

Das aufgeführte XML Statement ist ein Beispiel für eine Stornierungs Transaktion mit dem minimalen Set an Befehlen:

```
<ns5:IPGApiOrderRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            mlns:ns4="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:Transaction>
            <ns4:PayPalTxType>
                  <ns4:Type>void</ns4:Type>
            </ns4:PayPalTxType>
            <ns4:TransactionDetails>
                  <ns4:OrderId>
                        C-32121f4d-852f-4f48-8095-8585b917c079
                  </ns4:OrderId>
                  <ns4:TDate>1321609898</ns4:TDate>
            </ns4:TransactionDetails>
      </ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

In case your system is not aware of the payment method that has been used for the original transaction, the Void can be performed using any TxType which supports Voids. The gateway will then select the correct payment method based on the referenced Order ID and TDate.

5.3.4 Gutschrift

Bitte beachten Sie, dass die Gutschrift transaction spezielle Nutzerrechte erfordert.

Das aufgeführte XML Statement ist ein Beispiel für eine Gutschrift Transaktion mit dem minimalen Set an Befehlen

```
<ns5:IPGApiOrderRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            mlns:ns4="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:Transaction>
            <ns4:PayPalTxType>
                  <ns4:Type>credit</ns4:Type>
            </ns4:PayPalTxType>
            <ns4:ClickandBuyData />
            <ns4:Payment>
                  <ns4:ChargeTotal>1</ns4:ChargeTotal>
```

```
                <ns4:Currency>EUR</ns4:Currency>
            </ns4:Payment>
            <ns4:Billing>
                <ns4:Email>x@y.zz</ns4:Email>
            </ns4:Billing>
        </ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

Anders als bei anderen Zahlarten enthalten PayPal Transaktionen keine Zahlinformationen wie Kartennummern enthalten. Deshalb erfordert diese Transaktion die registrierte Mail-Adresse des Zahlungsempfängers. Die Mail-Adresse ist in dem Feld ns4:Billing/ns4:Email einzutragen.

5.4 SOFORT Überweisung

5.4.1 Return

Das folgende XML Statement zeigt beispielhaft eine *Return* transaction:

```
<ns5:IPGApiOrderRequest
            xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            mlns:ns4="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:Transaction>
            <ns4:SofortTxType>
                  <ns4:Type>return</ns4:Type>
            </ns4:SofortTxType>
            <ns4:Payment>
                  <ns4:ChargeTotal>1.00</ns4:ChargeTotal>
                  <ns4:Currency>EUR</ns4:Currency>
            </ns4:Payment>
            <ns4:TransactionDetails>
                  <ns4:OrderId>
                        C-32121f4d-852f-4f48-8095-8585b917c079
                  </ns4:OrderId>
            </ns4:TransactionDetails>
      </ns4:Transaction>
</ns5:IPGApiOrderRequest>
```

Für den Fall, dass Ihr System die Zahlmethode nicht kennen sollte, die für die ursprüngliche Transaktion genutzt wurde, kann jeder beliebige TxType für eine Return Transaktion genutzt werden.

Das Gateway wird in diesem Fall die passende Zahlart basierend auf der referenzierten OrderID auswählen.

6 Zusätzliche Aktionen über den Web-Service

6.1 Manueller Kassenschnitt

Entsprechend einer Zahlungstransaktion kann auch ein manueller Kassenschnitt über den Web-Service angestoßen werden:

```
<ipgapi:IPGApiActionRequest
            xmlns:a1="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <a1:Action>
            </a1:InitiateClearing>
      </a1:Action>
</ipgapi:IPGApiActionRequest>
```

Der Kassenschnitt wird hiermit initiiert und innerhalb der nächsten Minuten durchgeführt.

Sollte der Kassenschnitt nicht ausgeführt werden können, sendet das Gateway den Hinweis „false“.

```
<ipgapi:IPGApiActionResponse
      xmlns:ipgapi=http://ipg-online.com/ipgapi/schemas/ipgapi
      xmlns:a1="http://ipg-online.com/ipgapi/schemas/a1"
      xmlns:pay_1_0_0="http://api.clickandbuy.com/webservices/pay_1_0_0/"
      xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">

      <ipgapi:successfully>false</ipgapi:successfully>

</ipgapi:IPGApiActionResponse>
```

6.2 Anfragen zu einer Bestellung (Inquiry Order)

Mit Hilfe des Action-Requests *InquiryOrder* können Sie Details zu erfolgten Transaktionen einer Bestellung abrufen. Übergeben Sie dazu die entsprechende Bestellnummer (Order ID):

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns2:Action>
            <ns2:InquiryOrder>
                  <ns2:OrderId>
                        b5b7fb49-3310-4212-9103-5da8bd026600
                  </ns2:OrderId>
            </ns2:InquiryOrder>
      </ns2:Action>
</ns4:IPGApiActionRequest>
```

Die Antwort enthält Informationen zu allen Transaktionen, die zu der übermittelten Order ID gehören:

```
<ns4:IPGApiActionResponse
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
      <ns4:OrderId>a058e6e1-fec7-4e63-a092-fab4ec689716</ns4:OrderId>
      <ns3:Billing>
            <ns3:CustomerID>customerID</ns3:CustomerID>
            <ns3:Name>äÄöÖüÜß</ns3:Name>
            <ns3:Company>company</ns3:Company>
            <ns3:Address1>address1</ns3:Address1>
            <ns3:Address2>address2</ns3:Address2>
            <ns3:City>city</ns3:City>
            <ns3:State>state</ns3:State>
            <ns3:Zip>zip</ns3:Zip>
            <ns3:Country>country</ns3:Country>
            <ns3:Phone>phone</ns3:Phone>
            <ns3:Fax>fax</ns3:Fax>
```

```
        <ns3:Email>email</ns3:Email>
</ns3:Billing>
<ns3:Shipping>
        <ns3:Name>shipping name</ns3:Name>
        <ns3:Address1>shipping address1</ns3:Address1>
        <ns3:Address2>shipping address2</ns3:Address2>
        <ns3:City>shipping city</ns3:City>
        <ns3:State>shipping state</ns3:State>
        <ns3:Zip>shipping zip</ns3:Zip>
        <ns3:Country>shipping country</ns3:Country>
</ns3:Shipping>
<ns2:TransactionValues>
        <ns3:CreditCardTxType>
                <ns3:Type>sale</ns3:Type>
        </ns3:CreditCardTxType>
        <ns3:CreditCardData>
                <ns3:CardNumber>5426...4979</ns3:CardNumber>
                <ns3:ExpMonth>12</ns3:ExpMonth>
                <ns3:ExpYear>12</ns3:ExpYear>
        </ns3:CreditCardData>
        <ns3:Payment>
                <ns3:ChargeTotal>1</ns3:ChargeTotal>
                <ns3:Currency>978</ns3:Currency>
        </ns3:Payment>
        <ns3:TransactionDetails>
                <ns3:InvoiceNumber>invoice number</ns3:InvoiceNumber>
                <ns3:OrderId>
                        a058e6e1-fec7-4e63-a092-fab4ec689716
                </ns3:OrderId>
                <ns3:Ip>127.0.0.1</ns3:Ip>
                <ns3:TDate>1273570668</ns3:TDate>
                <ns3:TransactionOrigin>ECI</ns3:TransactionOrigin>
        </ns3:TransactionDetails>
        <ns3:Billing>
                <ns3:CustomerID>customerID</ns3:CustomerID>
        </ns3:Billing>
        <ns4:IPGApiOrderResponse>
                <ns4:ApprovalCode>
                        Y:356887:0000144820:PPXM:0612789753
                </ns4:ApprovalCode>
                <ns4:AVSResponse>PPX</ns4:AVSResponse>
                <ns4:OrderId>
                        a058e6e1-fec7-4e63-a092-fab4ec689716
                </ns4:OrderId>
                <ns4:ProcessorApprovalCode>
                        356887
                </ns4:ProcessorApprovalCode>
                <ns4:ProcessorReceiptNumber>
                        9753
                </ns4:ProcessorReceiptNumber>
                <ns4:ProcessorTraceNumber>
                        061278
                </ns4:ProcessorTraceNumber>
                <ns4:TDate>1273570668</ns4:TDate>
                <ns4:TDateFormatted>
                        2010.05.11 11:37:48 (MESZ)
                </ns4:TDateFormatted>
                <ns4:TerminalID>54000668</ns4:TerminalID>
        </ns4:IPGApiOrderResponse>
        <ns2:ReceiptNumber>9753</ns2:ReceiptNumber>
        <ns2:TraceNumber>61278</ns2:TraceNumber>
        <ns2:TransactionState>VOIDED</ns2:TransactionState>
</ns2:TransactionValues>
<ns2:TransactionValues>
        <ns3:CreditCardTxType>
                <ns3:Type>credit</ns3:Type>
        </ns3:CreditCardTxType>
        <ns3:CreditCardData>
```

```
                <ns3:CardNumber>5426...4979</ns3:CardNumber>
                <ns3:ExpMonth>12</ns3:ExpMonth>
                <ns3:ExpYear>12</ns3:ExpYear>
        </ns3:CreditCardData>
        <ns3:Payment>
                <ns3:ChargeTotal>1</ns3:ChargeTotal>
                <ns3:Currency>978</ns3:Currency>
        </ns3:Payment>
        <ns3:TransactionDetails>
                <ns3:InvoiceNumber>invoice number</ns3:InvoiceNumber>
                <ns3:OrderId>
                        a058e6e1-fec7-4e63-a092-fab4ec689716
                </ns3:OrderId>
                <ns3:Ip>127.0.0.1</ns3:Ip>
                <ns3:TDate>1273570670</ns3:TDate>
                <ns3:TransactionOrigin>ECI</ns3:TransactionOrigin>
        </ns3:TransactionDetails>
        <ns3:Billing>
                <ns3:CustomerID>customerID</ns3:CustomerID>
        </ns3:Billing>
        <ns4:IPGApiOrderResponse>
                <ns4:ApprovalCode>
                        Y:533118:0000144821:PPXM:0160050943
                </ns4:ApprovalCode>
                <ns4:AVSResponse>PPX</ns4:AVSResponse>
                <ns4:OrderId>
                        a058e6e1-fec7-4e63-a092-fab4ec689716
                </ns4:OrderId>
                <ns4:ProcessorApprovalCode>
                        533118
                </ns4:ProcessorApprovalCode>
                <ns4:ProcessorReceiptNumber>
                        0943
                </ns4:ProcessorReceiptNumber>
                <ns4:ProcessorTraceNumber>
                        016005
                </ns4:ProcessorTraceNumber>
                <ns4:TDate>1273570670</ns4:TDate>
                <ns4:TDateFormatted>
                        2010.05.11 11:37:50 (MESZ)
                </ns4:TDateFormatted>
                <ns4:TerminalID>54000669</ns4:TerminalID>
        </ns4:IPGApiOrderResponse>
        <ns2:ReceiptNumber>943</ns2:ReceiptNumber>
        <ns2:TraceNumber>16005</ns2:TraceNumber>
        <ns2:TransactionState>CAPTURED</ns2:TransactionState>
</ns2:TransactionValues>
<ns2:TransactionValues>
        <ns3:CreditCardTxType>
                <ns3:Type>void</ns3:Type>
        </ns3:CreditCardTxType>
        <ns3:CreditCardData>
                <ns3:CardNumber>5426...4979</ns3:CardNumber>
                <ns3:ExpMonth>12</ns3:ExpMonth>
                <ns3:ExpYear>12</ns3:ExpYear>
        </ns3:CreditCardData>
        <ns3:Payment>
                <ns3:ChargeTotal>1</ns3:ChargeTotal>
                <ns3:Currency>978</ns3:Currency>
        </ns3:Payment>
        <ns3:TransactionDetails>
                <ns3:InvoiceNumber>invoice number</ns3:InvoiceNumber>
                <ns3:OrderId>
                        a058e6e1-fec7-4e63-a092-fab4ec689716
                </ns3:OrderId>
                <ns3:Ip>127.0.0.1</ns3:Ip>
                <ns3:TDate>1273570672</ns3:TDate>
                <ns3:TransactionOrigin>ECI</ns3:TransactionOrigin>
```

```
            </ns3:TransactionDetails>
            <ns3:Billing>
                  <ns3:CustomerID>customerID</ns3:CustomerID>
            </ns3:Billing>
            <ns4:IPGApiOrderResponse>
                  <ns4:ApprovalCode>
                        Y:356887:0000144820:PPX :0612799754
                  </ns4:ApprovalCode>
                  <ns4:AVSResponse>PPX</ns4:AVSResponse>
                  <ns4:OrderId>
                        a058e6e1-fec7-4e63-a092-fab4ec689716
                  </ns4:OrderId>
                  <ns4:ProcessorApprovalCode>
                        356887
                  </ns4:ProcessorApprovalCode>
                  <ns4:ProcessorReceiptNumber>
                        9754
                  </ns4:ProcessorReceiptNumber>
                  <ns4:ProcessorTraceNumber>
                        061279
                  </ns4:ProcessorTraceNumber>
                  <ns4:TDate>1273570672</ns4:TDate>
                  <ns4:TDateFormatted>
                        2010.05.11 11:37:52 (MESZ)
                  </ns4:TDateFormatted>
                  <ns4:TerminalID>54000668</ns4:TerminalID>
            </ns4:IPGApiOrderResponse>
            <ns2:ReceiptNumber>9754</ns2:ReceiptNumber>
            <ns2:TraceNumber>61279</ns2:TraceNumber>
      </ns2:TransactionValues>
      <ns2:TransactionValues>
            <ns3:CreditCardTxType>
                  <ns3:Type>postauth</ns3:Type>
            </ns3:CreditCardTxType>
            <ns3:CreditCardData />
            <ns3:Payment>
                  <ns3:ChargeTotal>1</ns3:ChargeTotal>
                  <ns3:Currency>978</ns3:Currency>
            </ns3:Payment>
            <ns3:TransactionDetails>
                  <ns3:OrderId>
                        a058e6e1-fec7-4e63-a092-fab4ec689716
                  </ns3:OrderId>
                  <ns3:Ip>127.0.0.1</ns3:Ip>
                  <ns3:TDate>1273570673</ns3:TDate>
                  <ns3:TransactionOrigin>ECI</ns3:TransactionOrigin>
            </ns3:TransactionDetails>
            <ns3:Billing>
                  <ns3:CustomerID>customerID</ns3:CustomerID>
            </ns3:Billing>
            <ns4:IPGApiOrderResponse>
                  <ns4:ApprovalCode>
                        N:-10501:invalid postauth attempt
                  </ns4:ApprovalCode>
                  <ns4:OrderId>
                        a058e6e1-fec7-4e63-a092-fab4ec689716
                  </ns4:OrderId>
                  <ns4:TDate>1273570673</ns4:TDate>
                  <ns4:TDateFormatted>
                        2010.05.11 11:37:53 (MESZ)
                  </ns4:TDateFormatted>
            </ns4:IPGApiOrderResponse>
            <ns2:TransactionState>DECLINED</ns2:TransactionState>
      </ns2:TransactionValues>
</ns4:IPGApiActionResponse>
```

6.3 Get Last Orders

Dieser Befehl stellt eine Abfrage über die letzten Bestellungen dar.

6.3.1 Latest orders of a Store

Abfrage über die letzten „n“ Bestellungen eines beliebigen Shops.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/v1" xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns3:Action>
                <ns3:GetLastOrders>
                    <ns3:Count>5</ns3:Count>
                </ns3:GetLastOrders>
            </ns3:Action>
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6.3.2 Latest orders of a Store within a given date range

Abfrage über die letzten “n” Bestellungen eines beliebigen Shops in einem definierten Zeitraum.

Beide Felder “DateFrom” und “DateTo” sind in dem Befehl xs:dateTime zu definieren.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/v1" xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns3:Action>
                <ns3:GetLastOrders>
                    <ns3:Count>5</ns3:Count>
                    <ns3:DateFrom>2014-04-05T10:23:37.143+02:00</ns3:DateFrom>
                    <ns3:DateTo>2014-05-05T10:23:37.143+02:00</ns3:DateTo>
```

```
            </ns3:GetLastOrders>
        </ns3:Action>
    </ns5:IPGApiActionRequest>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6.3.3 Alle Bestellungen eines Shops für eine bestimmte OrderID

Dieser Befehl unterstützt eine umfangreiche Abfrage von durchgeführten Bestellungen eines bestimmten Shops in einem definierten Zeitraum

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/v1" xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns3:Action>
                <ns3:GetLastOrders>
                    <ns3:Count>2</ns3:Count>
                    <ns3:OrderID>Test SGSDAO.ConversionDate
1382020873203</ns3:OrderID>
                </ns3:GetLastOrders>
            </ns3:Action>
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope
```

6.3.4 Response

Alle Abfragemethoden liefern Antworten mit derselben Struktur zurück:

- Der Erfolgsstatus wird folgendermaßen zurückgeliefert
  `<ipgapi:successfully>true</ipgapi:successfully>`
- `<ipgapi:ResultInfo>/`
  `<a1:MoreResultsAvailable>true</a1:MoreResultsAvailable>`
  Diese Antwort drückt aus, dass mehrere Ergebisse verfügbar sind:
  - The service is stateless, therefore subsequent queries for pagination have to use either…
  - GetLastOrders(storeID, count, dateFrom, dateTo) w/ dateTo set to the last order's order_date of the previous resultset OR

- GetLastOrders(storeID, count, orderId) w/ orderId set to the last order of the previous resultset

- List of orders `<ipgapi:OrderValues>`, consisting of
  - `OrderId` – the orders' unique id
  - `<a1:TransactionValues>` transactions
  - `<v1:Basket>` the basket
    - with basket-items `<v1:Item>`
    - and each item with item-options `<v1:Option>`

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ipgapi:IPGApiActionResponse xmlns:ipgapi="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:a1="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:pay_1_0_0="http://api.clickandbuy.com/webservices/pay_1_0_0/"
xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">
            <ipgapi:successfully>true</ipgapi:successfully>
            <ipgapi:ResultInfo>
                <a1:MoreResultsAvailable>true</a1:MoreResultsAvailable>
            </ipgapi:ResultInfo>
            <ipgapi:OrderValues>
                <a1:OrderId>API-Test b9dfe057-7df9-4b46-bd40-9879adec3289
InquiryOrderTest::testBasket(305)</a1:OrderId>
                <a1:OrderDate>2013-10-17T14:37:06.000+02:00</a1:OrderDate>
                <v1:Basket>
                    <v1:Item>
                        <v1:ID>ID</v1:ID>
                        <v1:Description>Description</v1:Description>
                        <v1:ChargeTotal>1</v1:ChargeTotal>
                        <v1:Quantity>13</v1:Quantity>
                        <v1:Option>
                            <v1:Name>Name 2</v1:Name>
                            <v1:Choice>Choice</v1:Choice>
                        </v1:Option>
                        <v1:Option>
                            <v1:Name>Name 1</v1:Name>
                            <v1:Choice>Choice</v1:Choice>
                        </v1:Option>
                    </v1:Item>
                    <v1:Item>
                        <v1:ID>ID</v1:ID>
                        <v1:Description>Description</v1:Description>
```

```
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Quantity>13</v1:Quantity>
                    <v1:Option>
                        <v1:Name>Name 2</v1:Name>
                        <v1:Choice>Choice</v1:Choice>
                    </v1:Option>
                    <v1:Option>
                        <v1:Name>Name 1</v1:Name>
                        <v1:Choice>Choice</v1:Choice>
                    </v1:Option>
                </v1:Item>
                <v1:Item>
                    <v1:ID>ID 3</v1:ID>
                    <v1:Description>Description 3</v1:Description>
                    <v1:ChargeTotal>3</v1:ChargeTotal>
                    <v1:Quantity>33</v1:Quantity>
                </v1:Item>
            </v1:Basket>
            <v1:Billing/>
            <v1:Shipping/>
            <a1:TransactionValues>
                <v1:CreditCardTxType>
                    <v1:Type>sale</v1:Type>
                </v1:CreditCardTxType>
                <v1:CreditCardData>
                    <v1:CardNumber>403587...4977</v1:CardNumber>
                    <v1:ExpMonth>12</v1:ExpMonth>
                    <v1:ExpYear>14</v1:ExpYear>
                    <v1:Brand>VISA</v1:Brand>
                </v1:CreditCardData>
                <v1:Payment>
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Currency>978</v1:Currency>
                </v1:Payment>
                <v1:TransactionDetails>
                    <v1:OrderId>API-Test b9dfe057-7df9-4b46-bd40-9879adec3289
InquiryOrderTest::testBasket(305)</v1:OrderId>
                    <v1:TDate>1382020626</v1:TDate>
                    <v1:TransactionOrigin>ECI</v1:TransactionOrigin>
                </v1:TransactionDetails>
```

```
                    <ipgapi:IPGApiOrderResponse>

<ipgapi:ApprovalCode>Y:194737:0095905551:PPXM:1338513274</ipgapi:ApprovalCode>
                        <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                        <ipgapi:Brand>VISA</ipgapi:Brand>
                        <ipgapi:OrderId>API-Test b9dfe057-7df9-4b46-bd40-
9879adec3289 InquiryOrderTest::testBasket(305)</ipgapi:OrderId>
                        <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>

<ipgapi:ProcessorApprovalCode>194737</ipgapi:ProcessorApprovalCode>

<ipgapi:ProcessorReceiptNumber>3274</ipgapi:ProcessorReceiptNumber>

<ipgapi:ProcessorCCVResponse>M</ipgapi:ProcessorCCVResponse>

<ipgapi:ProcessorTraceNumber>133851</ipgapi:ProcessorTraceNumber>
                        <ipgapi:TDate>1382020626</ipgapi:TDate>
                        <ipgapi:TDateFormatted>2013.10.17 16:37:06
(MESZ)</ipgapi:TDateFormatted>
                        <ipgapi:TerminalID>54000668</ipgapi:TerminalID>
                    </ipgapi:IPGApiOrderResponse>
                    <a1:TransactionState>CAPTURED</a1:TransactionState>
                    <a1:UserID>1</a1:UserID>
                    <a1:SubmissionComponent>API</a1:SubmissionComponent>
                </a1:TransactionValues>
            </ipgapi:OrderValues>
            <ipgapi:OrderValues>
                <a1:OrderId>API-Test 7c59f1cf-24e3-48ae-b3fa-e7daed007564
InquiryOrderTest::testBasket(305)</a1:OrderId>
                <a1:OrderDate>2013-10-17T14:35:26.000+02:00</a1:OrderDate>
                <v1:Basket>
                    <v1:Item>
                        <v1:ID>ID</v1:ID>
                        <v1:Description>Description</v1:Description>
                        <v1:ChargeTotal>1</v1:ChargeTotal>
                        <v1:Quantity>13</v1:Quantity>
                        <v1:Option>
                            <v1:Name>Name 2</v1:Name>
                            <v1:Choice>Choice</v1:Choice>
                        </v1:Option>
                        <v1:Option>
                            <v1:Name>Name 1</v1:Name>
```

```
                <v1:Choice>Choice</v1:Choice>
            </v1:Option>
        </v1:Item>
        <v1:Item>
            <v1:ID>ID</v1:ID>
            <v1:Description>Description</v1:Description>
            <v1:ChargeTotal>1</v1:ChargeTotal>
            <v1:Quantity>13</v1:Quantity>
            <v1:Option>
                <v1:Name>Name 2</v1:Name>
                <v1:Choice>Choice</v1:Choice>
            </v1:Option>
            <v1:Option>
                <v1:Name>Name 1</v1:Name>
                <v1:Choice>Choice</v1:Choice>
            </v1:Option>
        </v1:Item>
        <v1:Item>
            <v1:ID>ID 3</v1:ID>
            <v1:Description>Description 3</v1:Description>
            <v1:ChargeTotal>3</v1:ChargeTotal>
            <v1:Quantity>33</v1:Quantity>
        </v1:Item>
    </v1:Basket>
    <v1:Billing/>
    <v1:Shipping/>
    <a1:TransactionValues>
        <v1:CreditCardTxType>
            <v1:Type>sale</v1:Type>
        </v1:CreditCardTxType>
        <v1:CreditCardData>
            <v1:CardNumber>403587...4977</v1:CardNumber>
            <v1:ExpMonth>12</v1:ExpMonth>
            <v1:ExpYear>14</v1:ExpYear>
            <v1:Brand>VISA</v1:Brand>
        </v1:CreditCardData>
        <v1:Payment>
            <v1:ChargeTotal>1</v1:ChargeTotal>
            <v1:Currency>978</v1:Currency>
```

```
                    </v1:Payment>
                    <v1:TransactionDetails>
                        <v1:OrderId>API-Test 7c59f1cf-24e3-48ae-b3fa-e7daed007564
InquiryOrderTest::testBasket(305)</v1:OrderId>
                        <v1:TDate>1382020526</v1:TDate>
                        <v1:TransactionOrigin>ECI</v1:TransactionOrigin>
                    </v1:TransactionDetails>
                    <ipgapi:IPGApiOrderResponse>

<ipgapi:ApprovalCode>Y:666159:0095905550:PPXM:0656200249</ipgapi:ApprovalCode>
                        <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                        <ipgapi:Brand>VISA</ipgapi:Brand>
                        <ipgapi:OrderId>API-Test 7c59f1cf-24e3-48ae-b3fa-
e7daed007564 InquiryOrderTest::testBasket(305)</ipgapi:OrderId>
                        <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>

<ipgapi:ProcessorApprovalCode>666159</ipgapi:ProcessorApprovalCode>

<ipgapi:ProcessorReceiptNumber>0249</ipgapi:ProcessorReceiptNumber>

<ipgapi:ProcessorCCVResponse>M</ipgapi:ProcessorCCVResponse>

<ipgapi:ProcessorTraceNumber>065620</ipgapi:ProcessorTraceNumber>
                        <ipgapi:TDate>1382020526</ipgapi:TDate>
                        <ipgapi:TDateFormatted>2013.10.17 16:35:26
(MESZ)</ipgapi:TDateFormatted>
                        <ipgapi:TerminalID>54000669</ipgapi:TerminalID>
                    </ipgapi:IPGApiOrderResponse>
                    <a1:TransactionState>CAPTURED</a1:TransactionState>
                    <a1:UserID>1</a1:UserID>
                    <a1:SubmissionComponent>API</a1:SubmissionComponent>
                </a1:TransactionValues>
            </ipgapi:OrderValues>
        </ipgapi:IPGApiActionResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6.4 Get Last Transactions

Dieser Befehl gibt Auskunft über die letzten Transaktionen

6.4.1 Die letzten Transaktionen eines Shops

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/a1" xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns2:Action>
                <ns2:GetLastTransactions>
                    <ns2:count>2</ns2:count>
                </ns2:GetLastTransactions>
            </ns2:Action>
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6.4.2 Alle Transaktionen eines Shops und einer definierten TransactionID

Eine transactionID besteht aus:

- `OrderId`
- `TDat,` Datum der Transaktion

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ns5:IPGApiActionRequest xmlns:ns5="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/a1" xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1"
xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
            <ns2:Action>
                <ns2:GetLastTransactions>
                    <ns2:count>2</ns2:count>
                    <ns2:OrderId>A-eb65437a-c538-4cdd-82b3-
d316ae160c22</ns2:OrderId>
                    <ns2:TDate>1407373211</ns2:TDate>
                </ns2:GetLastTransactions>
            </ns2:Action>
```

```
        </ns5:IPGApiActionRequest>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6.4.3 Response

Alle Antworten dieser Abfragen weisen dieselbe Struktur auf:

- Erfolgsstatus: `<ipgapi:successfully>true</ipgapi:successfully>`
- Auflistung der Transaktionen `<a1:TransactionValues>`

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
    <SOAP-ENV:Header/>
    <SOAP-ENV:Body>
        <ipgapi:IPGApiActionResponse xmlns:ipgapi="http://ipg-
online.com/ipgapi/schemas/ipgapi" xmlns:a1="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:pay_1_0_0="http://api.clickandbuy.com/webservices/pay_1_0_0/"
xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">
            <ipgapi:successfully>true</ipgapi:successfully>
            <a1:TransactionValues>
                <v1:CreditCardTxType>
                    <v1:Type>periodic</v1:Type>
                </v1:CreditCardTxType>
                <v1:CreditCardData>
                    <v1:CardNumber>403587...4977</v1:CardNumber>
                    <v1:ExpMonth>12</v1:ExpMonth>
                    <v1:ExpYear>14</v1:ExpYear>
                    <v1:Brand>VISA</v1:Brand>
                </v1:CreditCardData>
                <v1:Payment>
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Currency>978</v1:Currency>
                </v1:Payment>
                <v1:TransactionDetails>
                    <v1:OrderId>A-bcbb36ad-90ad-4ff7-ad96-
b5d73dd9c5e9</v1:OrderId>
                    <v1:TDate>1407373210</v1:TDate>
                </v1:TransactionDetails>
                <ipgapi:IPGApiOrderResponse>

<ipgapi:ApprovalCode>Y:272450:0014750514:PPXM:0433836659</ipgapi:ApprovalCode>
```

```
                    <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                    <ipgapi:Brand>VISA</ipgapi:Brand>
                    <ipgapi:OrderId>A-bcbb36ad-90ad-4ff7-ad96-
b5d73dd9c5e9</ipgapi:OrderId>
                    <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>

<ipgapi:ProcessorApprovalCode>272450</ipgapi:ProcessorApprovalCode>

<ipgapi:ProcessorReceiptNumber>6659</ipgapi:ProcessorReceiptNumber>
                    <ipgapi:ProcessorCCVResponse>M</ipgapi:ProcessorCCVResponse>

<ipgapi:ProcessorTraceNumber>043383</ipgapi:ProcessorTraceNumber>
                    <ipgapi:ReferencedTDate>1407373210</ipgapi:ReferencedTDate>
                    <ipgapi:TDate>1407373210</ipgapi:TDate>
                    <ipgapi:TDateFormatted>2014.08.07 03:00:10
(CEST)</ipgapi:TDateFormatted>
                    <ipgapi:TerminalID>54000667</ipgapi:TerminalID>
                </ipgapi:IPGApiOrderResponse>
                <a1:TransactionState>CAPTURED</a1:TransactionState>
                <a1:UserID>1</a1:UserID>
                <a1:SubmissionComponent>BUS</a1:SubmissionComponent>
            </a1:TransactionValues>
            <a1:TransactionValues>
                <v1:CreditCardTxType>
                    <v1:Type>periodic</v1:Type>
                </v1:CreditCardTxType>
                <v1:CreditCardData>
                    <v1:CardNumber>403587...4977</v1:CardNumber>
                    <v1:ExpMonth>12</v1:ExpMonth>
                    <v1:ExpYear>14</v1:ExpYear>
                    <v1:Brand>VISA</v1:Brand>
                </v1:CreditCardData>
                <v1:Payment>
                    <v1:ChargeTotal>1</v1:ChargeTotal>
                    <v1:Currency>978</v1:Currency>
                </v1:Payment>
                <v1:TransactionDetails>
                    <v1:OrderId>A-52421c39-69c4-4b2d-959d-
9fdcd3a9420a</v1:OrderId>
                    <v1:TDate>1407373209</v1:TDate>
                </v1:TransactionDetails>
```

```
                <ipgapi:IPGApiOrderResponse>

<ipgapi:ApprovalCode>Y:416502:0014750513:PPXM:4625106408</ipgapi:ApprovalCode>
                    <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                    <ipgapi:Brand>VISA</ipgapi:Brand>
                    <ipgapi:OrderId>A-52421c39-69c4-4b2d-959d-
9fdcd3a9420a</ipgapi:OrderId>
                    <ipgapi:PaymentType>CREDITCARD</ipgapi:PaymentType>

<ipgapi:ProcessorApprovalCode>416502</ipgapi:ProcessorApprovalCode>

<ipgapi:ProcessorReceiptNumber>6408</ipgapi:ProcessorReceiptNumber>
                    <ipgapi:ProcessorCCVResponse>M</ipgapi:ProcessorCCVResponse>

<ipgapi:ProcessorTraceNumber>462510</ipgapi:ProcessorTraceNumber>
                    <ipgapi:ReferencedTDate>1407373209</ipgapi:ReferencedTDate>
                    <ipgapi:TDate>1407373209</ipgapi:TDate>
                    <ipgapi:TDateFormatted>2014.08.07 03:00:09
(CEST)</ipgapi:TDateFormatted>
                    <ipgapi:TerminalID>54000666</ipgapi:TerminalID>
                </ipgapi:IPGApiOrderResponse>
                <a1:TransactionState>CAPTURED</a1:TransactionState>
                <a1:UserID>1</a1:UserID>
                <a1:SubmissionComponent>BUS</a1:SubmissionComponent>
            </a1:TransactionValues>
        </ipgapi:IPGApiActionResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

6.5 Wiederkehrende Zahlungen

Der Action-Request *RecurringPayment* ermöglicht Ihnen die Einrichtung von regelmäßigen Abbuchungen von einem Kreditkartenkonto. Daneben kann auch eine einzelne Transaktion vorgemerkt werden, die automatisch zu einem späteren Zeitpunkt ausgeführt wird. Grundsätzlich stehen drei Funktionen zur Verfügung **install**, **modify** und **cancel**.

Das folgende Beispiel zeigt Einrichtung einer einmaligen Zahlung, die zum 31.12.2011 ausgeführt werden soll:

```
<ns4:IPGApiActionRequest
            xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns2:Action>
            <ns2:RecurringPayment>
                  <ns2:Function>install</ns2:Function>
```

```
                        <ns2:RecurringPaymentInformation>
                                <ns2:RecurringStartDate>
                                        20111231
                                </ns2:RecurringStartDate>
                                <ns2:InstallmentCount>1</ns2:InstallmentCount>
                                <ns2:InstallmentFrequency>
                                        1
                                </ns2:InstallmentFrequency>
                                <ns2:InstallmentPeriod>
                                        month
                                </ns2:InstallmentPeriod>
                        </ns2:RecurringPaymentInformation>
                        <ns2:CreditCardData>
                                <ns3:CardNumber>4035875676474977</ns3:CardNumber>
                                <ns3:ExpMonth>12</ns3:ExpMonth>
                                <ns3:ExpYear>12</ns3:ExpYear>
                                <ns3:CardCodeValue>977</ns3:CardCodeValue>
                        </ns2:CreditCardData>
                        <ns3:Payment>
                                <ns3:ChargeTotal>1</ns3:ChargeTotal>
                                <ns3:Currency>978</ns3:Currency>
                        </ns3:Payment>
                </ns2:RecurringPayment>
        </ns2:Action>
</ns4:IPGApiActionRequest>
```

Wird als Startdatum (*RecurringStartDate*) das aktuelle Datum angegeben, wird die erste Zahlungstransaktion sofort angestoßen. Die Angabe eines Datums in der Vergangenheit ist nicht erlaubt.

Der für *TransactionOrigin* verwendete Standardwert ist ‚ECI'. Möchten Sie einen anderen Wert festlegen, können Sie diesen als TransactionOrigin im RecurringPayment Tag übergeben.

Eine Änderung einer wiederkehrenden Zahlung kann unter Angabe der dazugehörigen *OrderId* vorgenommen werden:

```
<ns4:IPGApiActionRequest
                xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
                xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
        <ns2:Action>
                <ns2:RecurringPayment>
                        <ns2:Function>modify</ns2:Function>
                        <ns2:OrderId>
                                e368a525-173f-4f56-9ae2-beb4023a6993
                        </ns2:OrderId>
                        <ns2:RecurringPaymentInformation>
                                <ns2:InstallmentCount>999</ns2:InstallmentCount>
                        </ns2:RecurringPaymentInformation>
                </ns2:RecurringPayment>
        </ns2:Action>
</ns4:IPGApiActionRequest>
```

Hierbei müssen nur die Angaben übergeben werden, die geändert werden sollen. Bei Änderung der Kreditkartennummer ist es dabei notwendig, auch das Verfalldatum zu übergeben. Bei Änderung des Betrags ist eine zusätzliche Angabe der Währung notwendig.

Die Löschung einer wiederkehrenden Zahlung erfolgt ebenfalls unter Angabe der *OrderId*:

```
<ns4:IPGApiActionRequest
                xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
                xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
        <ns2:Action>
                <ns2:RecurringPayment>
```

```
                    <ns2:Function>cancel</ns2:Function>
                    <ns2:OrderId>
                          e368a525-173f-4f56-9ae2-beb4023a6993
                    </ns2:OrderId>
              </ns2:RecurringPayment>
       </ns2:Action>
</ns4:IPGApiActionRequest>
```

Die erfolgreiche Einrichtung, Änderung oder Löschung wiederkehrender Zahlungen wird mit **true** bestätigt:

```
<ns4:IPGApiActionResponse
              xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
              xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
              xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
       <ns4:successfully>true</ns4:successfully>
       <ns4:OrderId>e368a525-173f-4f56-9ae2-beb4023a6993</ns4:OrderId>
</ns4:IPGApiActionResponse>
```

6.6 Validierung von Kreditkartendaten

Mit dem Action-Request ‘Validate’ können Sie Kreditkartendaten validieren. Dabei wird im Hintergrund eine Vorautorisierung auf das Kreditkartenkonto in Höhe von 1 EUR durchgeführt und im Anschluss sofort wieder storniert.

```
<ns4:IPGApiActionRequest
                    xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                    xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
                    xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
       <ns3:Action>
              <ns3:Validate>
                    <ns3:CreditCardData>
                          <ns2:CardNumber>403587567644977</ns2:CardNumber>
                          <ns2:ExpMonth>12</ns2:ExpMonth>
                          <ns2:ExpYear>12</ns2:ExpYear>
                    </ns3:CreditCardData>
              </ns3:Validate>
       </ns3:Action>
</ns4:IPGApiActionRequest>
```

In der Antwort erhalten Sie alle Informationen zu den beiden Vorgängen. Die entscheidende Information finden Sie im Tag ‚successfully’ - steht hier der Wert *true*, konnten die Kartendaten erfolgreich validiert werden:

```
<ns4:IPGApiActionResponse
                    xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                    xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
                    xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
       <ns4:successfully>true</ns4:successfully>
       <ns2:TransactionValues>
              <ns3:CreditCardTxType>
                    <ns3:Type>preauth</ns3:Type>
              </ns3:CreditCardTxType>
              <ns3:Payment>
                    <ns3:ChargeTotal>1</ns3:ChargeTotal>
                    <ns3:Currency>978</ns3:Currency>
              </ns3:Payment>
              <ns3:TransactionDetails>
                    <ns3:OrderId>
                          A-5337067b-33cb-4173-b003-129ccf437c0c
                    </ns3:OrderId>
                    <ns3:TDate>1250599046</ns3:TDate>
              </ns3:TransactionDetails>
              <ns4:IPGApiOrderResponse>
```

```
                <ns4:ApprovalCode>
                        Y:66923202:0000075790:PPXM:0011511144
                </ns4:ApprovalCode>
                <ns4:AVSResponse>PPX</ns4:AVSResponse>
                <ns4:CommercialServiceProvider>
                        BNLP
                </ns4:CommercialServiceProvider>
                <ns4:OrderId>
                        A-5337067b-33cb-4173-b003-129ccf437c0c
                </ns4:OrderId>
                <ns4:ProcessorApprovalCode>
                        66923202
                </ns4:ProcessorApprovalCode>
                <ns4:ProcessorReceiptNumber>
                        1144
                </ns4:ProcessorReceiptNumber>
                <ns4:ProcessorReferenceNumber>
                        66923202
                </ns4:ProcessorReferenceNumber>
                <ns4:ProcessorResponseCode>00</ns4:ProcessorResponseCode>
                <ns4:ProcessorResponseMessage>
                        Function performed error-free
                </ns4:ProcessorResponseMessage>
                <ns4:ProcessorTraceNumber>
                        001151
                </ns4:ProcessorTraceNumber>
                <ipgapi:TDate>1330682626</ipgapi:TDate>
                <ipgapi:TDateFormatted>
                        2012.03.02 11:03:46 (MEZ)
                </ipgapi:TDateFormatted>
                <ns4:TerminalID>99000003</ns4:TerminalID>
                <ns4:TransactionResult>APPROVED</ns4:TransactionResult>
                <ns4:TransactionTime>1250599048</ns4:TransactionTime>
        </ns4:IPGApiOrderResponse>
        <ns2:ResponseCode>00</ns2:ResponseCode>
</ns2:TransactionValues>
<ns2:TransactionValues>
        <ns3:CreditCardTxType>
                <ns3:Type>void</ns3:Type>
        </ns3:CreditCardTxType>
        <ns3:TransactionDetails>
                <ns3:OrderId>
                        A-5337067b-33cb-4173-b003-129ccf437c0c
                </ns3:OrderId>
                <ns3:TDate>1250599046</ns3:TDate>
        </ns3:TransactionDetails>
        <ns4:IPGApiOrderResponse>
                <ns4:ApprovalCode>
                        Y:9451977:0000075790:PPX :0011521145
                </ns4:ApprovalCode>
                <ns4:AVSResponse>PPX</ns4:AVSResponse>
                <ns4:CommercialServiceProvider>
                        BNLP
                </ns4:CommercialServiceProvider>
                <ns4:OrderId>
                        A-5337067b-33cb-4173-b003-129ccf437c0c
                </ns4:OrderId>
                <ns4:ProcessorApprovalCode>
                        9451977
                </ns4:ProcessorApprovalCode>
                <ns4:ProcessorReceiptNumber>
                        1145
                </ns4:ProcessorReceiptNumber>
                <ns4:ProcessorReferenceNumber>
                        9451977
                </ns4:ProcessorReferenceNumber>
                <ns4:ProcessorResponseCode>00</ns4:ProcessorResponseCode>
                <ns4:ProcessorResponseMessage>
```

```
                        Function performed error-free
                    </ns4:ProcessorResponseMessage>
                    <ns4:ProcessorTraceNumber>
                        001152
                    </ns4:ProcessorTraceNumber>
                    <ns4:TDate>1250599046</ns4:TDate>
                    <ns4:TerminalID>99000003</ns4:TerminalID>
                    <ns4:TransactionResult>APPROVED</ns4:TransactionResult>
                    <ns4:TransactionTime>1250599049</ns4:TransactionTime>
            </ns4:IPGApiOrderResponse>
            <ns2:ResponseCode>00</ns2:ResponseCode>
      </ns2:TransactionValues>
</ns4:IPGApiActionResponse>
```

Im negativen Fall enthält das Tag ‘successfully’ den Wert *false* und eine Fehlermeldung (‚ErrorMessage’), die Auskunft darüber gibt, warum die Karte abgelehnt wurde:

```
<ns4:IPGApiActionResponse
                    xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                    xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
                    xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>false</ns4:successfully>
      <ns2:Error Code="SGS-002303">
            <ns2:ErrorMessage>
                  Invalid credit card number.
            </ns2:ErrorMessage>
      </ns2:Error>
      <ns2:TransactionValues>
            <ns3:CreditCardTxType>
                  <ns3:Type>preauth</ns3:Type>
            </ns3:CreditCardTxType>
            <ns3:Payment>
                  <ns3:ChargeTotal>1</ns3:ChargeTotal>
                  <ns3:Currency>978</ns3:Currency>
            </ns3:Payment>
            <ns3:TransactionDetails>
                  <ns3:OrderId>
                        A-65aacf74-72ac-4684-b447-abb6d54ea5e4
                  </ns3:OrderId>
            </ns3:TransactionDetails>
            <ns4:IPGApiOrderResponse>
                  <ns4:ErrorMessage>
                        Invalid credit card number.
                  </ns4:ErrorMessage>
                  <ns4:OrderId>
                        A-65aacf74-72ac-4684-b447-abb6d54ea5e4
                  </ns4:OrderId>
                  <ns4:TransactionTime>1250599045</ns4:TransactionTime>
            </ns4:IPGApiOrderResponse>
      </ns2:TransactionValues>
</ns4:IPGApiActionResponse>
```

Erfolgreich überprüfte Kreditkarten können im selben Weg gespeichert werden, wie Sale Transaktionen. Details finden Sie im Kapitel Erweiterungsfunktion Data Storage.

6.7 Transaktionsstatus

Einige Payment Endpoints senden das Transaktionsergebnis nicht in Ihrer Antwort mit. In diesen Fällen sendet das IPG einen Bestätigungscode der mit einem Fragzeichen beginnt (?:..)

Der Befehl „GetExternalTransactionState“ erlaubt Ihnen den Status solcher Transaktionen abzufragen.

6.8 Anfrage von email notifications

Der Befehl “SendEMailNotification” stößt eine Email Benachrichtigung an für eine definierte Transaktion an.

Weitere Informationen zum Thema Email Benachrichtigungen finden Sie im Benutzerhandbuch,

```
<ns5:IPGApiActionRequest
            xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns2:Action>
            <ns2:SendEMailNotification>
                  <ns2:OrderId>0/8/15</ns2:OrderId>
                  <ns2:TDate>1250599046</ns2:TDate>
            </ns2:SendEMailNotification>
      </ns2:Action>
</ns5:IPGApiActionRequest>
```

6.9 Warenkorbinformationen und Produktkatalog

6.9.1 Warenkorbinformationen in Transaktionsbenachrichtigungen

Das folgende Beispiel zeigt wie Warenkorbinformationen genutzt werden können um Details über die Verkäufe zu dokumentieren.

```
<ns5:IPGApiOrderRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns2:Transaction>
            <ns2:CreditCardTxType>
```

```
            <ns2:Type>sale</ns2:Type>
        </ns2:CreditCardTxType>
        <ns2:CreditCardData>
            <ns2:CardNumber>403587XXXXXX4977</ns2:CardNumber>
            <ns2:ExpMonth>12</ns2:ExpMonth>
            <ns2:ExpYear>14</ns2:ExpYear>
        </ns2:CreditCardData>
        <ns2:Payment>
            <ns2:ChargeTotal>1</ns2:ChargeTotal>
            <ns2:Currency>EUR</ns2:Currency>
        </ns2:Payment>
        <ns2:TransactionDetails>
            <ns2:OrderId>68d4a595-fd58-4859-83cd-1ae13962a3ac</ns2:OrderId>
        </ns2:TransactionDetails>
        <ns2:Basket>
            <ns2:Item>
                <ns2:ID>product ID xyz</ns2:ID>
                <ns2:Description>description of abc</ns2:Description>
                <ns2:ChargeTotal>11</ns2:ChargeTotal>
                <ns2:Currency>EUR</ns2:Currency>
                <ns2:Quantity>5</ns2:Quantity>
                <ns2:Option>
                    <ns2:Name>colour</ns2:Option>
                    <ns2:Choice>blue</ns2:Choice>
                </ns2:Option>
                <ns2:Option>
                    <ns2:Name>size</ns2:Option>
                    <ns2:Choice>large</ns2:Choice>
                </ns2:Option>
            </ns2:Item>
        </ns2:Basket>
    </ns2:Transaction>
</ns5:IPGApiOrderRequest>
```

6.9.2 Einen Produktkatalog aufsetzen

Basisinformationen über Ihre Produkte können folgendermaßen mitgeliefert werden:

```
<ns5:IPGApiActionRequest
        xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
        xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
```

```
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns3:Action>
            <ns3:ManageProducts>
                  <ns3:Function>store</ns3:Function>
                  <ns3:Product>
                        <ns3:ProductID>product ID xyz</ns3:ProductID>
                        <ns2:ChargeTotal>2</ns2:ChargeTotal>
                        <ns2:Currency>EUR</ns2:Currency>
                        <ns3:OfferStarts>
                              2014-12-27T13:29:41.000+01:00
                        </ns3:OfferStarts>
                        <ns3:OfferEnds>
                              2015-09-19T14:29:41.000+02:00
                        </ns3:OfferEnds>
                        <ns2:Option>
                              <ns2:Name>colour</ns2:Option>
                              <ns2:Choice>blue</ns2:Choice>
                        </ns2:Option>
                        <ns2:Option>
                              <ns2:Name>size</ns2:Option>
                              <ns2:Choice>large</ns2:Choice>
                        </ns2:Option>
                  </ns3:Product>
            </ns3:ManageProducts>
      </ns3:Action>
</ns5:IPGApiActionRequest>
```

OfferStarts und OfferEnds sind optional und können dazu genutzt werden, um die Sichtbarkeit von Produkten in der Kundenanwendung einzuschränken, ohne dass dies Einfluss auf den Verkauf der Produkte hätte. Es gibt weitere optionale Feldbeschreibungen wie Description, OptionName und Name. Weitere Details finden Sie im Anhang a1.xsd dieses Dokumentes.

```
<ns5:IPGApiActionRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns3:Action>
```

```
            <ns3:ManageProducts>
                  <ns3:Function>display</ns3:Function>
                  <ns3:Product>
                        <ns3:ProductID>product ID xyz</ns3:ProductID>
                  </ns3:Product>
            </ns3:ManageProducts>
      </ns3:Action>
</ns5:IPGApiActionRequest>
```

Die delete Funktion kann genutzt werden, um den Lagerbestand eines Produktes auf Null zu setzen.

```
<ns5:IPGApiActionRequest
            xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns3:Action>
            <ns3:ManageProducts>
                  <ns3:Function>delete</ns3:Function>
                  <ns3:Product>
                        <ns3:ProductID>product ID xyz</ns3:ProductID>
                  </ns3:Product>
            </ns3:ManageProducts>
      </ns3:Action>
</ns5:IPGApiActionRequest>
```

6.9.3 Den Lagerbestand managen

Nach der Einrichtung eines Poduktes, können Sie mit der Funktion “Add”, Details zum Lagerbestand ergänzen.

```
<ns5:IPGApiActionRequest
            xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi
            xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
            xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
      <ns3:Action>
            <ns3:ManageProductStock>
                  <ns3:Function>add</ns3:Function>
```

```
                <ns3:ProductStock>
                        <ns3:ProductID>product ID xyz</ns3:ProductID>
                        <ns2:Option>
                                <ns2:Name>colour</ns2:Option>
                                <ns2:Choice>blue</ns2:Choice>
                        </ns2:Option>
                        <ns2:Option>
                                <ns2:Name>size</ns2:Option>
                                <ns2:Choice>large</ns2:Choice>
                        </ns2:Option>
                        <ns3:Quantity>13</ns3:Quantity>
                </ns3:ProductStock>
        </ns3:ManageProductStock>
    </ns3:Action>
</ns5:IPGApiActionRequest>
```

6.9.4 Sale Transaktionen und Lagerbestand

Nachdem Sie den Lagerbestand eingerichtet haben, können Sie diesen für die Prüfung nutzen, ob der Lagerbestand für Ihre Verkäufe ausreicht, Sollte der Lagerbestand für den anstehenden Verkauf nicht genügen, wird die Transaktion abgelehnt.

Um diese Funktionalität zu nutzen, fügen Sie diesen Befehl hinzu:

add **`<ns2:ProductStock>check</ns2:ProductStock>`** to Basket.

```
<ns5:IPGApiOrderRequest
        xmlns:ns5="http://ipg-online.com/ipgapi/schemas/ipgapi"
        xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
        xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1"
        xmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
    <ns2:Transaction>
        <ns2:CreditCardTxType>
            <ns2:Type>sale</ns2:Type>
        </ns2:CreditCardTxType>
        <ns2:CreditCardData>
            <ns2:CardNumber>403587XXXXXX4977</ns2:CardNumber>
            <ns2:ExpMonth>12</ns2:ExpMonth>
            <ns2:ExpYear>14</ns2:ExpYear>
        </ns2:CreditCardData>
        <ns2:Payment>
```

```
                <ns2:ChargeTotal>1</ns2:ChargeTotal>
                <ns2:Currency>EUR</ns2:Currency>
            </ns2:Payment>
            <ns2:TransactionDetails>
                <ns2:OrderId>68d4a595-fd58-4859-83cd-1ae13962a3ac</ns2:OrderId>
            </ns2:TransactionDetails>
            <ns2:Basket>
                <ns2:ProductStock>check</ns2:ProductStock>
                <ns2:Item>
                    <ns2:ID>product ID xyz</ns2:ID>
                    <ns2:Description>description of abc</ns2:Description>
                    <ns2:ChargeTotal>11</ns2:ChargeTotal>
                    <ns2:Currency>EUR</ns2:Currency>
                    <ns2:Quantity>5</ns2:Quantity>
                    <ns2:Option>
                        <ns2:Name>colour</ns2:Option>
                        <ns2:Choice>blue</ns2:Choice>
                    </ns2:Option>
                    <ns2:Option>
                        <ns2:Name>size</ns2:Option>
                        <ns2:Choice>large</ns2:Choice>
                    </ns2:Option>
                </ns2:Item>
            </ns2:Basket>
        </ns2:Transaction>
</ns5:IPGApiOrderRequest>
```

7 Erweiterungsfunktion Data Storage

Mit der Erweiterungsfunktion Data Storage können Sie Kreditkartendaten und Bankverbindungen Ihrer Kunden bei First Data Austria in einer verschlüsselten Datenbank hinterlegen, um auf diese für spätere Zahlungen zurückgreifen zu können.

Wenn Sie diese Funktion beauftragt haben, bietet Ihnen First Data Austria API die in den folgenden Abschnitten beschriebenen Funktionen.

Weitere Möglichkeiten der Erweiterungsfunktion Data Storage finden Sie im Integrationsleitfaden First Data Austria Connect.

7.1 Speicherung/Aktualisierung von Zahlungsinformationen bei einer Transaktion

Übermitteln Sie bei einer Zahlungstransaktion zusätzlich den Parameter *HostedDataID* mit einer eindeutigen Kennung für die zu hinterlegenden Zahlungsinformationen. Je nach Zahlart werden unter dieser Kennung Kartennummer und Verfalldatum bzw. Kontonummer und Bankleitzahl hinterlegt. Ist die ‚HostedDataID' bereits vorhanden, werden die gespeicherten Informationen überschrieben.

```
<ipgapi:IPGApiOrderRequest
      xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
      xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
      <v1:Transaction>
            <v1:CreditCardTxType>
                  <v1:Type>sale</v1:Type>
            </v1:CreditCardTxType>
            <v1:CreditCardData>
                  <v1:CardNumber>4111111111111111</v1:CardNumber>
                  <v1:ExpMonth>12</v1:ExpMonth>
                  <v1:ExpYear>07</v1:ExpYear>
            </v1:CreditCardData>
            <v1:Payment>
                  <v1:HostedDataID>
                        HDID customer 1234567
                  </v1:HostedDataID>
                  <v1:ChargeTotal>19.00</v1:ChargeTotal>
                  <v1:Currency>978</v1:Currency>
            </v1:Payment>
      </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

Die Speicherung des Datensatzes erfolgt nur im Falle einer erfolgreich autorisierten Zahlungstransaktion und wenn Ihr Shop für die Nutzung dieser Funktion freigeschaltet wurde.

7.2 Zahlungstransaktionen mit hinterlegten Daten

Wurden zu einem Kunden Zahlungsinformationen hinterlegt, können Sie unter Angaben der ‚HostedDataID' Zahlungstransaktionen durchführen ohne dass eine erneute Angabe von Kartennummer und Verfalldatum bzw. Kontonummer und Bankleitzahl notwendig sind.

Bitte beachten Sie, dass die Kartenprüfnummer (meistens auf der Rückseite der Kreditkarte) nicht gespeichert werden darf und daher bei Kreditkartenzahlungen zusätzlich vom Kunden angegeben werden muss. Für den Zahlungsablauf im Shop empfiehlt es sich in Ihrem System z. B. auch die letzten vier Stellen der hinterlegten Kreditkartennummer vorzuhalten, sodass Sie diese dem Kunden anzeigen können und er weiß, welche seiner ggf. mehreren Karten bei Ihnen für die Zahlung registriert ist.

```
<ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
```

```
        <v1:Transaction>
                <v1:CreditCardTxType>
                        <v1:Type>sale</v1:Type>
                </v1:CreditCardTxType>
                <v1:Payment>
                        <v1:HostedDataID>HDID customer 1234567</v1:HostedDataID>
                        <v1:ChargeTotal>19.00</v1:ChargeTotal>
                        <v1:Currency>978</v1:Currency>
                </v1:Payment>
        </v1:Transaction>
</ipgapi:IPGApiOrderRequest>
```

7.3 Hinterlegen eines Datensatzes ohne gleichzeitige Zahlungstransaktion

Neben der Möglichkeit, neue Datensätze im Zuge einer Zahlungstransaktion zu hinterlegen, können Sie auch Hosted Data IDs über einen Action-Request hinterlegen. Dabei ist es auch möglich, mehrere Datensätze gleichzeitig zur Speicherung zu übertragen. Das folgende Beispiel zeigt die Hinteregung eines Datensatzes mit Kreditkartendaten sowie eines Eintrags mit Kontonummer und Bankleitzahl. Bitte beachten Sie, dass auch hier gilt: wird eine bereits vorhandene Hosted Data ID übermittelt, wird der bisher vorhandene Datensatz aktualisiert.

```
<ns4:IPGApiActionRequest
                xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
                xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
        <ns2:Action>
                <ns2:StoreHostedData>
                        <ns2:DataStorageItem>
                                <ns2:CreditCardData>
                                        <ns3:CardNumber>
                                                4035875676474977
                                        </ns3:CardNumber>
                                        <ns3:ExpMonth>12</ns3:ExpMonth>
                                        <ns3:ExpYear>08</ns3:ExpYear>
                                </ns2:CreditCardData>
                                <ns2:HostedDataID>
                                        d763bba7-1cfa-4d3d-94af-9fbe29ec0e26
                                </ns2:HostedDataID>
                        </ns2:DataStorageItem>
                        <ns2:DataStorageItem>
                                <ns2:DE_DirectDebitData>
                                        <ns3:BankCode>50014560</ns3:BankCode>
                                        <ns3:AccountNumber>
                                                32121503
                                        </ns3:AccountNumber>
                                </ns2:DE_DirectDebitData>
                                <ns2:HostedDataID>
                                        691c7cb3-a752-4d6d-abde-83cad63de258
                                </ns2:HostedDataID>
                        </ns2:DataStorageItem>
                </ns2:StoreHostedData>
        </ns2:Action>
</ns4:IPGApiActionRequest>
```

Bei erfolgreicher Hinterlegung erhalten Sie folgende Bestätigung:

```
<ns4:IPGApiActionResponse
                xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
                xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
        <ns4:successfully>true</ns4:successfully>
</ns4:IPGApiActionResponse>
```

Konnten eine oder mehrere Datensätze nicht hinterlegt werden, werden die betroffenen Hosted Data IDs in der Antwort entsprechend gekennzeichnet:

```
<ns4:IPGApiActionResponse
        xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
        xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
        xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
    <ns4:successfully>true</ns4:successfully>
    <ns2:Error Code="SGSDAS-020300">
        <ns2:ErrorMessage>
            Could not store the hosted data id:
            691c7cb3-a752-4d6d-abde-83cad63de258.
            Reason: An internal error has occured while processing
            your request
        </ns2:ErrorMessage>
    </ns2:Error>
</ns4:IPGApiActionResponse>
```

7.4 Sperrung der Verwendung von gleichen Zahlungsinformationen für mehrere Datensätze

Zur Vermeidung, dass Kunden sich bei Ihnen mehrmals mit denselben Zahlungsinformationen registrieren können, besteht die Möglichkeit, bei der Speicherung das zusätzliche Tag ***DeclineHostedDataDuplicates*** zu übermitteln. Gültige Werte für diesen Parameter sind ‘true’ und ’false’. Wird der Wert ‘true’ übergeben und die übermittelten Zahlungsinformationen liegen bereits unter einer anderen ‘hosteddataid’ vor, wird der Vorgang abgelehnt.

7.5 Abruf hinterlegter Datensätze

Mit Hilfe des Action-Requests ‚display’, haben Sie die Möglichkeit, hinterlegte Datensätze unter der Angabe der entsprechenden Hosted Data ID abzurufen:

```
<ns4:IPGApiActionRequest
        xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
        xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
        xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
    <ns3:Action>
        <ns3:StoreHostedData>
            <ns3:DataStorageItem>
                <ns3:Function>display</ns3:Function>
                <ns3:HostedDataID>
                    d56feaaf-2d96-4159-8fd6-887e07fc9052
                </ns3:HostedDataID>
            </ns3:DataStorageItem>
        </ns3:StoreHostedData>
    </ns3:Action>
</ns4:IPGApiActionRequest>
```

Die Antwort beinhaltet die hinterlegten Zahlungsinformationen. Von Kreditkartennummern werden dabei aus Sicherheitsgründen nur die ersten 6 und letzten 4 Stellen ausgegeben:

```
<ns4:IPGApiActionResponse
        xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
        xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
        xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
    <ns4:successfully>true</ns4:successfully>
    <ns4:DataStorageItem>
        <ns2:CreditCardData>
            <ns3:CardNumber>403587XXXXXX4977</ns3:CardNumber>
            <ns3:ExpMonth>12</ns3:ExpMonth>
```

```
                        <ns3:ExpYear>12</ns3:ExpYear>
                    </ns2:CreditCardData>
                    <ns2:HostedDataID>
                        d56feaaf-2d96-4159-8fd6-887e07fc9052
                    </ns2:HostedDataID>
                </ns4:DataStorageItem>
        </ns4:IPGApiActionResponse>
```

Existiert die von Ihnen übermittelte Hosted Data ID nicht, erhalten Sie eine entsprechende Fehlermeldung:

```
        <ns4:IPGApiActionResponse
                    xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                    xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
                    xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
                <ns4:successfully>true</ns4:successfully>
                <ns2:Error Code="SGSDAS-020301">
                    <ns2:ErrorMessage>
                        Hosted data id:
                        6c814261-a843-49fb-bacd-1411d3780286 not found.
                    </ns2:ErrorMessage>
                </ns2:Error>
        </ns4:IPGApiActionResponse>
```

7.6 Löschen hinterlegter Datensätze

Über den Action-Request ‚delete' können Sie nicht mehr benötigte Datensätze löschen:

```
<ns4:IPGApiActionRequest
                    xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                    xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
                    xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
        <ns3:Action>
                <ns3:StoreHostedData>
                    <ns3:DataStorageItem>
                        <ns3:Function>delete</ns3:Function>
                        <ns3:HostedDataID>
                            9605c2d1-428c-4de2-940e-4bec4737ab5d
                        </ns3:HostedDataID>
                    </ns3:DataStorageItem>
                </ns3:StoreHostedData>
        </ns3:Action>
</ns4:IPGApiActionRequest>
```

Eine erfolgreiche Löschung wird mit folgender Antwort bestätigt:

```
<ns4:IPGApiActionResponse
                    xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                    xmlns:ns2="http://ipg-online.com/ipgapi/schemas/a1"
                    xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
        <ns4:successfully>true</ns4:successfully>
</ns4:IPGApiActionResponse>
```

7.7 Prüfen bestehender Datensatze

Gespeicherte Kreditkartendetails können mit der Aktion „Validate" überprüft werden.

```
<ns5:IPGApiActionRequest
xmlns:ns5=http://ipg-online.com/ipgapi/schemas/ipgapi xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/a1"
xmlns:ns3="hmlns:ns4="http://api.clickandbuy.com/webservices/pay_1_0_0/">
        <ns2:Action>
                <ns2:Validate>
```

```
                    <ns3:Payment>
                          <ns3:HostedDataID>
      5937c993-8011-46e6-b9bc-fac07eff4b50
      </ns3:HostedDataID>
                    </ns3:Payment>
               </ns2:Validate>
      </ns2:Action>
</ns5:IPGApiActionRequest>
```

Eine Prüfung deutscher Debitkarten wird nicht unterstützt und wird mit einer Fehlermeldung beantwortet.

7.8 Löschen bestehender Datensatze

Mit der Aktion „Delete“ können Datensätze, die nicht länger benötigt werden, gelöscht werden.

```
<ns4:IPGApiActionRequest
xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/v1" xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
      <ns3:Action>
            <ns3:StoreHostedData>
                  <ns3:DataStorageItem>
                        <ns3:Function>delete</ns3:Function>
                        <ns3:HostedDataID>
      9605c2d1-428c-4de2-940e-4bec4737ab5d
      </ns3:HostedDataID>
                  </ns3:DataStorageItem>
            </ns3:StoreHostedData>
      </ns3:Action>
</ns4:IPGApiActionRequest>
```

Bei erfolgreicher Löschung erhalten Sie folgende Antwort:

```
<ns4:IPGApiActionResponse
xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi" xmlns:ns2="http://ipg-
online.com/ipgapi/schemas/a1" xmlns:ns3="http://ipg-online.com/ipgapi/schemas/v1">
      <ns4:successfully>true</ns4:successfully>
</ns4:IPGApiActionResponse>
```

8 Erweiterungsfunktion Pay&Call

Mit der Erweiterungsfunktion Pay&Call können Sie Prepaid-Konten verschiedener Mobilfunkprovider über das First Data Austria Internet Payment Gateway aufladen.

First Data Austria API bietet dazu den Transaktionstyp „MPCharge“, bei dem der Netzbetreiber, die Mobilfunknummer und der entsprechende Ladebetrag übermittelt werden:

```
<ns4:IPGApiOrderRequest
                    xmlns:ns4="http://ipg-online.com/ipgapi/schemas/ipgapi"
                    xmlns:ns2="http://ipg-online.com/ipgapi/schemas/v1"
                    xmlns:ns3="http://ipg-online.com/ipgapi/schemas/a1">
      <ns2:Transaction>
            <ns2:TopUpTxType>
                  <ns2:MPCharge>
                        <ns2:MNSP>D1</ns2:MNSP>
                        <ns2:MSISDN>01707616513</ns2:MSISDN>
                        <ns2:PaymentType>JCB</ns2:PaymentType>
                  </ns2:MPCharge>
            </ns2:TopUpTxType>
            <ns2:Payment>
                  <ns2:ChargeTotal>15</ns2:ChargeTotal>
                  <ns2:Currency>EUR</ns2:Currency>
            </ns2:Payment>
      </ns2:Transaction>
</ns4:IPGApiOrderRequest>
```

Eine Beschreibung der verwendeten Elemente sowie weiterer optionaler Elemente finden Sie im Kapitel **Übersicht der XML-Tags**.

9 Übersicht der XML-Tags

9.1 Übersicht nach Transaktionstyp

Die folgende Tabelle zeigt für jeden Transaktionstyp, welche XML-Tags übermittelt werden müssen und welche optional verwendet werden können. Bitte verwenden Sie ausschließlich hier angegebene Felder und beachten Sie die dargestellte Reihenfolge.

Verwendete Abkürzungen:

m: Pflichtfeld

o: optional

d: optional mit Vorgabewert

a und **b**: maximal eine der beiden Angaben

1: wenn **a** oder b angegeben wurde optional, Pflichtfeld wenn **a** und **b** nicht angegeben wurden

3: Pflichtfeld für 3D Secure Transaktionen

s: Nähere Details im Abschnitt CreditCard3DSecure

Pfad/ Name	**Kreditkarte**							**Pay& Call**
alle Pfadangaben relativ zu `ipgapi:IPGApiOrderRequest/ v1:Transaction`	Sale	ForceTicket	PreAuth	PostAuth	Return	Credit	Void	MPCharge
v1:CreditCardTxType/ v1:Type	**m**	**m**	**m**	**m**	**m**	**m**	**m**	
v1:CreditCardData/ v1:CardNumber	**a**	**a**	**a**			**a**		
v1:CreditCardData/ v1:ExpMonth	**a**	**a**	**a**			**a**		
v1:CreditCardData/ v1:ExpYear	**a**	**a**	**a**			**a**		
v1:CreditCardData/ v1:CardCodeValue	*o*	*o*	*o*			*o*		
v1:CreditCardData/ v1:TrackData	**B**	**b**	**b**			**b**		
v1:CreditCardData/ v1:Brand	*o*	*o*	*o*			*O*		
v1:CreditCard3DSecure/ v1:VerificationResponse	3	3	3			3		
v1:CreditCard3DSecure/ v1:PayerAuthenticationResponse	S	s	s			s		
v1:CreditCard3DSecure/ v1:AuthenticationValue	S	s	s			s		

v1:CreditCard3DSecure/ v1:XID	S	s	s			s		
v1:ClickandBuyTxType/ v1:Type								
v1:ClickandBuyData/ cab:OrderDetails								
v1:Payment/ v1:HostedDataID	**1**	**1**	**1**			**1**		
v1:Payment/ v1:HostedDataStoreID	**1**	**1**	**1**			**1**		
v1:Payment/ v1:DeclineHostedDataDuplicates	**1**	**1**	**1**			**1**		
v1:Payment/ v1:SubTotal	*o*	*o*	*o*	*o*	*o*	*o*		
v1:Payment/ v1:ValueAddedTax	*o*	*o*	*o*	*o*	*o*	*o*		
v1:Payment/ v1:DeliveryAmount	*o*	*o*	*o*	*o*	*o*	*o*		
v1:Payment/ v1:ChargeTotal	**m**	**m**	**m**	**m**	**m**	**m**		**M**
v1:Payment/ v1:Currency	**m**	**m**	**m**	**m**	**m**	**m**		**M**
v1:TransactionDetails/ v1:OrderId	*o*	*o*	*o*	**m**	**m**	*o*	**m**	*O*
v1:TransactionDetails/ v1:Ip	*o*		*o*			*o*		*O*
v1:TransactionDetails/ v1:ReferenceNumber		**m**						
v1:TransactionDetails/ v1:Tdate							**m**	
v1:TransactionDetails/ v1:TransactionOrigin	*d*		*d*			*d*		
v1:TransactionDetails/ v1:InvoiceNumber	*o*	*o*	*o*			*o*		*O*
v1:TransactionDetails/ v1:PONumber	*o*	*o*	*o*			*o*		*O*
v1:TransactionDetails/ v1:DynamicMerchantNumber	*o*	*o*	*o*			*o*		*O*
v1:TransactionDetails/ v1:Comments	*o*	*o*	*o*	*o*	*o*	*o*	*o*	*O*
v1:TransactionDetails/ v1:Terminal/ v1:TerminalID	*o*	*o*	*o*			*o*		
v1:TransactionDetails/ v1:InquiryRateReference	*o*	*o*	*o*					
v1:Billing/ v1:CustomerID	*o*	*o*	*o*			*o*		*O*

v1:Billing/ v1:Name	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:Company	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:Address1	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:Address2	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:City	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:State	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:Zip	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:Country	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:Phone	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:Fax	*o*	*o*	*o*			*o*		*O*
v1:Billing/ v1:Email	*o*	*o*	*o*			*o*		*O*
v1:Shipping/ v1:Type	*o*	*o*	*o*			*o*		
v1:Shipping/ v1:Name	*o*	*o*	*o*			*o*		
v1:Shipping/ v1:Address1	*o*	*o*	*o*			*o*		
v1:Shipping/ v1:Address2	*o*	*o*	*o*			*o*		
v1:Shipping/ v1:City	*o*	*o*	*o*			*o*		
v1:Shipping/ v1:State	*o*	*o*	*o*			*o*		
v1:Shipping/ v1:Zip	*o*	*o*	*o*			*o*		
v1:Shipping/ v1:Country	*o*	*o*	*o*			*o*		
v1:TopUpTxType/ v1:MPCharge/ v1:MNSP								
v1:TopUpTxType/ v1:MPCharge/ v1:MSISDN								
v1:TopUpTxType/ v1:MPCharge/ v1:PaymentType								

v1:ClientLocale/ v1:Language	d	d	d	d	d	d	d	D
v1:ClientLocale/ v1:Country	d	d	d	d	d	d	d	D

Path/ **Name**	**ClickandBuy**				**PayPal**				**Mobile Top-up**

all paths relative to
`ipgapi:IPGApiOrderRequest/`
`v1:Transaction`
	PostAuth	Return	Credit	Void	PostAuth	Return	Gutschrift	Stornierung	MPCharge

v1:CreditCardTxType/ v1:Type									
v1:CreditCardData/ v1:CardNumber									
v1:CreditCardData/ v1:ExpMonth									
v1:CreditCardData/ v1:ExpYear									
v1:CreditCardData/ v1:CardCodeValue									
v1:CreditCardData/ v1:TrackData									
v1:CreditCard3DSecure/ v1:VerificationResponse									
v1:CreditCard3DSecure/ v1:PayerAuthenticationResponse									
v1:CreditCard3DSecure/ v1:AuthenticationValue									
v1:CreditCard3DSecure/ v1:XID									
v1:ClickandBuyTxType/ v1:Type	**m**	**m**	**m**	**m**					
v1:ClickandBuyData/ cab:OrderDetails	*o*	*o*	***a***	*o*					
V1: PayPalTxType V1:Type					**m**	**m**	**m**	**m**	
v1:Payment/ v1:HostedDataID									
v1:Payment/ v1:HostedDataStoreID									
v1:Payment/ v1:DeclineHostedDataDuplicates									

v1:Payment/ v1:ChargeTotal	**m**	**m**	**m**		**m**	**m**	**m**		**m**
v1:Payment/ v1:Currency	**m**	**m**	**m**		**m**	**m**	**m**		**m**
v1:TransactionDetails/ v1:OrderId	**m**	**m**	*o*	**m**	*m*	**m**	*o*	**m**	*o*
v1:TransactionDetails/ v1:Ip			*o*				*o*		*o*
v1:TransactionDetails/ v1:ReferenceNumber									
v1:TransactionDetails/ v1:Tdate				**m**				**m**	
v1:TransactionDetails/ v1:TransactionOrigin			*d*				**d**		
v1:TransactionDetails/ v1:InvoiceNumber			*o*				*o*		*o*
v1:TransactionDetails/ v1:PONumber			*o*				*o*		*o*
v1:TransactionDetails/ v1:DynamicMerchantName			*o*				*o*		*o*
v1:TransactionDetails/ v1:Comments	*o*	*o*	*o*	*o*	*o*	**o**	*o*	**o**	*o*
v1:Billing/ v1:CustomerID			*o*				*o*		*o*
v1:Billing/ v1:Name			*o*				*o*		*o*
v1:Billing/ v1:Company			*o*				*o*		*o*
v1:Billing/ v1:Address1			*o*				*o*		*o*
v1:Billing/ v1:Address2			*o*				*o*		*o*
v1:Billing/ v1:City			*o*				*o*		*o*
v1:Billing/ v1:State			*o*				*o*		*o*
v1:Billing/ v1:Zip			*o*				*o*		*o*
v1:Billing/ v1:Country			*o*				*o*		*o*
v1:Billing/ v1:Phone			*o*				*o*		*o*
v1:Billing/ v1:Fax			*o*				*o*		*o*
v1:Billing/ v1:Email			***1***				*o*		*o*

v1:Shipping/ v1:Type			*o*				*o*		
v1:Shipping/ v1:Name			*o*				*o*		
v1:Shipping/ v1:Address1			*o*				*o*		
v1:Shipping/ v1:Address2			*o*				*o*		
v1:Shipping/ v1:City			*o*				*o*		
v1:Shipping/ v1:State			*o*				*o*		
v1:Shipping/ v1:Zip			*o*				*o*		
v1:Shipping/ v1:Country			*o*				*o*		
v1:TopUpTxType/ v1:MPCharge/ v1:MNSP									**m**
v1:TopUpTxType/ v1:MPCharge/ v1:MSISDN									**m**
v1:TopUpTxType/ v1:MPCharge/ v1:PaymentType									**m**
v1:ClientLocale/ v1:Language	*d*	*d*	*d*	*d*	**d**	**d**	**d**	**d**	*d*
v1:ClientLocale/ v1:Country	*d*	*d*	*d*	*d*	**d**	**d**	**d**	**d**	*d*

9.2 Beschreibungen der XML-Tags

9.2.1 CreditCardTxType

Pfad / Name	XML Schematyp	Beschreibung
`v1:CreditCardTxType/` `v1:Type`	`xs:string`	Speichert den Transaktionstyp. Mögliche Werte sind `sale`, `forceTicket`, `preAuth`, `postAuth`, `return`, `credit` and `void`.

9.2.2 CreditCardData

Pfad / Name	XML Schematyp	Beschreibung
`v1:CreditCardData/ v1:CardNumber`	`xs:string`	Speichert die Kreditkartennummer des Kunden. Vergewissern Sie sich, dass diese Zeichenfolge nur Ziffern enthält. Die Übermittlung der Nummer beispielsweise im Format *xxxx-xxxx-xxxx-xxxx* würde zur Rückgabe eines Fehlers durch den First Data Austria API Web Service führen.
`v1:CreditCardData/ v1:ExpMonth`	`xs:string`	Speichert den Ablaufmonat der Kreditkarte des Kunden. Vergewissern Sie sich, dass dieses Element immer genau zwei Ziffern enthält. Eine Karte mit Ablaufdatum im Juli würde bedeuten, dass das dieses Element den Wert `07` enthält.
`v1:CreditCardData/ v1:ExpYear`	`xs:string`	Speichert das Ablaufjahr der Kreditkarte des Kunden. Es gelten die gleichen Formatvorschriften wie für das Element `v1:ExpMonth`.
`v1:CreditCardData/ v1:CardCodeValue`	`xs:string`	Speichert die drei- oder vierstellige Kartenprüfnummer (CVC, CVV). Sie ist meist auf der Rückseite der Karte aufgedruckt.
`v1:CreditCardData/ v1:TrackData`	`xs:string`	Speichert die Spurdaten des Magnetstreifens bei Erfassung der Kartendaten mittels Kartenleser (kann optional an Stelle von `CardNumber`, `ExpMonth` und `ExpYear` übergeben werden). Hierbei müssen mindestens die zusammenhängenden Daten aus Spur 1 und 2 inkl. der Trenner (`%…?;…?`) übergeben werden. Spur 3 Daten sind optional.
`v1:CreditCardData/ v1:TrackData`	`xs:string`	Optionales Feld für die Bezeichnung der Kreditkarte. Bei Nutzung des Feldes wird die Transaktion dann ausgeführt, wenn die Kartennummer mit dem Brand übereinstimmt.

9.2.3 CreditCard3DSecure

Pfad / Name	XML Schematyp	Beschreibung
`v1:CreditCard3DSecure/ v1:VerificationResponse`	`xs:string`	Speichert die VerificationResponse (VERes) des Merchant Plug-in.
`v1:CreditCard3DSecure/ v1:PayerAuthenticationResponse`	`xs:string`	Speichert die PayerAuthenticationResponse (PARes) des Merchant Plug-in.

v1:CreditCard3DSecure/ v1:AuthenticationValue	xs:string	Speichert die AuthenticationValue (MasterCard: AAV or VISA: CAAV) des Merchant Plug-in.
v1:CreditCard3DSecure/ v1:XID	xs:string	Speichert die XID des Merchant Plug-in.

Bitte beachten Sie, dass es sich hierbei um Werte handelt, die Sie ggf. von einem eigenen Merchant Plug-in oder einer Lösung eines 3D Secure Anbieters erhalten. Die Verwendung der Erweiterungsfunktion Online Secure ist aus technischen Gründen für Transaktionen über die API Lösung **nicht** möglich.

9.2.4 ClickandBuyTxType

Path/Name	XML Schema type	Description
v1:ClickandBuyTxType/ v1:Type	xs:string	Stores the transaction type. Possible values are postAuth, return, credit and void.

9.2.5 ClickandBuyData

Path/Name	XML Schema type	Description
v1:ClickandBuyData/ cab:OrderDetails	Xs:complexType	Stores detailed order information. See http://integration.clickandbuy-community.com/ or the ClickandBuy xsd which is part of the IPG API wsdl: PayTypes_1_0_0.xsd, SOAPTypes_1_0_0.xsd

9.2.6 PayPalTxType

Path/Name	XML Schema type	Beschreibung
v1:PayPalTxType/ v1:Type	xs:string	Speichert die Transaktionsarten: Reservierung, Rückzahlung Gutschrift und Storno.

9.2.7 Payment

Pfad / Name	XML Schematyp	Beschreibung
v1:Payment/ v1:HostedDataID	xs:string	Speichert die Hosted Data ID für die Erweiterungsfunktion Data Storage.
v1:Payment/ v1:HostedDataStoreID	xs:string	Speichert die Hosted Data Store ID für die Erweiterungsfunktion Data Storage (nur als technischer User)
v1:Payment/	xs:string	Ermöglicht das Sperren der Verwendung

v1:DeclineHostedDataDuplicates		von gleichen Zahlungsinformationen für mehrere Datensätze bei der Erweiterungsfunktion Data Storage
v1:Payment/ v1:ChargeTotal	xs:double	Speichert den Betrag der Transaktion. Die Nachkommastellen werden durch einen . separiert. Vergewissern Sie sich, dass die Zahl der Nachkommastellen 2 nicht überschreitet, z. B. wäre 3.123 ungültig, 3.12, 3.1 und 3 sind hingegen korrekt.
v1:Payment/ v1:Currency	xs:string	Speichert die Währung als dreistelligen ISO 4217 Wert (z. B. 978 für Euro)

9.2.8 TransactionDetails

Pfad / Name	XML Schematyp	Beschreibung
v1:TransactionDetails/ v1:OrderId	xs:string	Speichert die Order ID, diese muss für die Store ID eindeutig sein. Falls keine angegeben wird, wird von First Data Austria ein Wert generiert und mit dem Ergebnis der Transaktion zurück geliefert.
v1:TransactionDetails/ v1:Ip	xs:string	Speichert die IP-Adresse des Kunden, die vom First Data Austria API Web Service zur Betrugsabwehr anhand der IP-Adresse genutzt werden kann. Vergewissern Sie sich, dass die IP-Adresse im Format *xxx.xxx.xxx.xxx* bereitgestellt wird. 128.0.10.2 wäre beispielsweise eine gültige IP.
v1:TransactionDetails/ v1:ReferenceNumber	xs:string	Speichert die Referenznummer, die bei einer *ForceTicket* Transaktion übergeben werden muss. Der Wert dieses Elements beginnt immer mit den drei Buchstaben ‚NEW', gefolgt von der sechsstelligen Autorisierungsnummer, die bei der telefonischen Autorsierung erhalten wurde.
v1:TransactionDetails/ v1:TDate	xs:string	Speichert das TDate der *Sale*, *PostAuth*, *ForceTicket*, *Return* oder *Credit* Transaktion, auf die sich diese *Void* Transaktion bezieht. Ein TDate Wert wird innerhalb der Antwort auf eine erfolgreiche Transaktion einer dieser fünf Typen zurückgeliefert. Bei Veranlassung einer *Void* Transaktion müssen Sie das TDate zusätzlich zur Order-ID übersenden, damit die ursprüngliche Transaktion eindeutig identifiziert werden kann.
v1:TransactionDetails/ v1:TransactionOrigin	xs:string	Speichert die Art des Kartendatenerhalts. Mögliche Werte sind ECI (über das Internet), MOTO (per Post oder telefonisch)

		und RETAIL (Karteninhaber war persönlich anwesend und Karte lag physisch vor).
v1:TransactionDetails/ v1:InvoiceNumber	xs:string	Speichert die Rechnungsnummer.
v1:TransactionDetails/ v1:PONumber	xs:string	Speichert die Bestellnummer
v1:TransactionDetails/ v1:DynamicMerchantName	xs:string	Speichert den Händlername
v1:TransactionDetails/ v1:Comments	xs:string	Speichert die Kommentare

9.2.9 Billing

Pfad / Name	XML Schematyp	Beschreibung
v1:Billing/ v1:CustomerID	xs:string	Speichert Ihre Kennung für den Kunden.
v1:Billing/ v1:Name	xs:string	Speichert den Namen des Kunden. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:Company	xs:string	Speichert den Firmennamen des Kunden. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:Address1	xs:string	Speichert die erste Zeile der Kundenadresse. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:Address2	xs:string	Speichert die zweite Zeile der Kundenadresse. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:City	xs:string	Speichert die Stadt des Kunden. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:State	xs:string	Speichert das Bundesland des Kunden. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:Zip	xs:string	Speichert die Postleitzahl des Kunden. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:Country	xs:string	Speichert das Land des Kunden. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:Phone	xs:string	Speichert die Telefonnummer des Kunden. Wenn bereitgestellt, erscheint die Angabe

		später in Ihren Transaktionsberichten.
v1:Billing/ v1:Fax	xs:string	Speichert die Faxnummer des Kunden. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Billing/ v1:Email	xs:string	Speichert die E-Mail-Adresse des Kunden. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten. Wenn Sie die eMail-Benachrichtigungsfunktion benutzen, wird diese eMail als Ziel-eMail genutzt.

9.2.10 Shipping

Pfad / Name	XML Schematyp	Beschreibung
v1:Shipping/ v1:Type	xs:string	Speichert die Art der Lieferung.
v1:Shipping/ v1:Name	xs:string	Speichert den Namen des Empfängers. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Shipping/ v1:Address1	xs:string	Speichert die erste Zeile der Empfängersadresse. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Shipping/ v1:Address2	xs:string	Speichert die zweite Zeile der Empfängersadresse. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Shipping/ v1:City	xs:string	Speichert die Stadt des Empfängers. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Shipping/ v1:State	xs:string	Speichert das Bundesland des Empfängers. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Shipping/ v1:Zip	xs:string	Speichert die Postleitzahl des Empfängers. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.
v1:Shipping/ v1:Country	xs:string	Speichert das Land des Empfängers. Wenn bereitgestellt, erscheint die Angabe später in Ihren Transaktionsberichten.

9.2.11 TopUpTxType

Pfad / Name	XML Schematyp	Beschreibung
v1:TopUpTxType/	xs:string	Speichert den Mobilfunknetzbetreiber. Folgende Werte sind gültig: D1 (T-Mobile),

v1:MPCharge/ v1:MNSP		D2 (Vodafone), EP (E-Plus), VI (O2).
v1:TopUpTxType/ v1:MPCharge/ v1:MSISDN	xs:string	Speichert die Mobilfunknummer, die für die Aufladung verwendet werden soll.
v1:TopUpTxType/ v1:MPCharge/ v1:PaymentType	xs:string	Speichert die Zahlungsweise (ggf. separate Transaktion). Gültige Werte sind: Amex, Cash (Barzahlung), Diners, ECMC (MasterCard), JCB, VISA.

9.2.12 ClientLocale

Pfad / Name	XML Schematyp	Beschreibung
v1:ClientLocale/ v1:Language	xs:string	Wenn Sie die eMail-Benachrichtigungs-funktion benutzen, wird diese Sprache für die Kundenbenachrichtigung genutzt. Mögliche Werte sind: de, en, it.
v1:ClientLocale/ v1:Country	xs:string	Spezifiziert die Sprachvarianten. Dieser Wert kann nur gestezt werden, wenn v1:Language gesetzt ist. Mögliche Werte sind: DE, GB, IT. Wenn Sie v1:Country nicht setzen, wird ein passender Wert gewählt.

Wenn Sie keine Sprachinformationen übergeben, wird die Spracheinstellung aus Ihrem Webshop für die eMail-Benachrichtigungen genutzt.

10 SOAP Anfragenachricht erstellen

Nach Erstellung der Transaktion in XML muss eine SOAP-Anfragenachricht erstellt werden. Sie beschreibt den Aufruf der Web Service Operation, die Sie durchführen möchten. Das bedeutet, die XML-kodierte Transaktion, die Sie wie im vorangehenden Kapitel beschrieben erstellt haben, entspricht dem Argument der Operation. Die SOAP-Anfragenachricht hingegen kodiert den eigentlichen Operationsaufruf.

Bitte beachten sie die korrekte **UTF-8** Kodierung, da ansonsten die Sonderzeichen und Umlaute nicht korrekt verarbeitet werden.

Die Zusammenstellung einer solchen SOAP-Anfragenachricht ist recht einfach. Die vollständige SOAP-Nachricht, in die eine XML *Sale* Transaktion wie in Kapitel 5.1.1 vorgestellt verpackt ist, sieht so aus:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <ipgapi:IPGApiOrderRequest
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1"
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
                  <v1:Transaction>
                        <v1:CreditCardTxType>
                              <v1:Type>sale</v1:Type>
                        </v1:CreditCardTxType>
                        <v1:CreditCardData>
                              <v1:CardNumber>
                                    4111111111111111
                              </v1:CardNumber>
                              <v1:ExpMonth>12</v1:ExpMonth>
                              <v1:ExpYear>07</v1:ExpYear>
                        </v1:CreditCardData>
                        <v1:Payment>
                              <v1:ChargeTotal>19.00</v1:ChargeTotal>
                              <v1:Currency>978</v1:Currency>
                        </v1:Payment>
                  </v1:Transaction>
            </ipgapi:IPGApiOrderRequest>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Kurz gesagt enthält die SOAP-Anfragenachricht einen SOAP-Envelope, der aus einem Header und einem Body besteht. Für den Aufruf des Web Service sind keine speziellen Header-Einträge erforderlich. Der SOAP-Body enthält wie oben gezeigt als Subelement das XML-Transaktionsdokument. Bitte beachten Sie, dass für Transaktionen eines anderen Typs als *Sale* keine weitergehenden Anforderungen bestehen. Damit sieht das allgemeine Format einer SOAP-Anfragenachricht unabhängig vom eigentlichen Transaktionstyp wie folgt aus:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <ipgapi:IPGApiOrderRequest
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi"
            xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1">
                  <v1:Transaction>
                        <!-- transaction content -->
                  </v1:Transaction>
            </ipgapi:IPGApiOrderRequest>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Wie Sie sicher bemerkt haben, gibt es keine spezifischen Einträge darüber, welche Web Service Operation aufgerufen werden soll. Tatsächlich weist First Data Austria API das Element
`ipgapi:IPGApiOrderRequest` automatisch der entsprechenden Web Service Operation zu.

10.1 Beschreibung der XML-Tags

10.1.1 CreditCardTxType

Path/Name	XML Schema type	Description
`v1:CreditCardTxType/` `v1:Type`	`xs:string`	Stores the transaction type. Possible values are `sale`, `forceTicket`, `preAuth`, `postAuth`, `return`, `credit` and `void`.

10.1.2 CreditCardData

Path/Name	XML Schema type	Description
`v1:CreditCardData/` `v1:CardNumber`	`xs:string`	Stores the customer's credit card number. Make sure that the string contains only digits, i.e. passing the number e.g. in the format *xxxx-xxxx-xxxx-xxxx* will result in an error returned by the Web Service API.
`v1:CreditCardData/` `v1:ExpMonth`	`xs:string`	Stores the expiration month of the customer's credit card. Make sure that the content of this element always contains two digits, i.e. a card expiring in July will have this element with value `07`.
`v1:CreditCardData/` `v1:ExpYear`	`xs:string`	Stores the expiration year of the customer's credit card. The same formatting restrictions as for the `v1:ExpMonth` element apply here.
`v1:CreditCardData/` `v1:CardCodeValue`	`xs:string`	Stores the three or four digit card security code (CSC) – sometimes also referred to as card verification value (CVV) or code (CVC) – which is typically printed on the back of the credit card. For information about the benefits of CSC contact support.
`v1:CreditCardData/` `v1:TrackData`	`xs:string`	Stores the track data of a card when using a card reader instead of keying in card data (can optionally be used instead of transmitting `CardNumber`, `ExpMonth` and `ExpYear`). This field needs to contain at least the concatenated track 1 and 2 data. Track data 3 is optional. The track data must include the track and field separators as they are stored on the card. Example for

		the track data separator from track data 1 and 2 without the data: %…?;…?
v1:CreditCardData/ v1:TrackData	xs:string	Optional field for the brand of the credit card. If this field is set, the transaction will only be processed if the card number matches the brand.

Zu XML Tags, die sich auf Card Present Transaktionen mit einem Kartenleser und PIN Eingabe Device beziehen, finden Sie weitere Details im Appendix dieses Dokuments.

10.1.3 CreditCard3DSecure

Path/Name	XML Schema type	Description
v1:CreditCard3DSecure/ v1:VerificationResponse	xs:string	Stores the VerificationResponse (VERes) of your Merchant Plug-in.
v1:CreditCard3DSecure/ v1:PayerAuthenticationResponse	xs:string	Stores the PayerAuthenticationResponse (PARes) of your Merchant Plug-in.
v1:CreditCard3DSecure/ v1:AuthenticationValue	xs:string	Stores the AuthenticationValue (MasterCard: AAV or VISA: CAAV) of your Merchant Plug-in.
v1:CreditCard3DSecure/ v1:XID	xs:string	Stores the XID of your Merchant Plug-in.

Bitte beachten Sie, dass die im Connect integrierte 3D Secure Funktionalität nicht für API Transaktionen verwendet werden können.

10.1.4 ClickandBuyTxType

Path/Name	XML Schema type	Description
v1:ClickandBuyTxType/ v1:Type	xs:string	Stores the transaction type. Possible values are postAuth, return, credit and void.

10.1.5 ClickandBuyData

Path/Name	XML Schema type	Description
v1:ClickandBuyData/ cab:OrderDetails	xs:complexType	Stores detailed order information. See http://integration.clickandbuy-community.com/ or the ClickandBuy xsd which is part of the IPG API wsdl: PayTypes_1_0_0.xsd, SOAPTypes_1_0_0.xsd

10.1.6 PayPalTxType

Path/Name	XML Schema type	Description
`v1:PayPalTxType/` `v1:Type`	`xs:string`	Stores the transaction type. Possible values are `postAuth`, `return`, `credit` and `void`.

10.1.7 Payment

Path/Name	XML Schema type	Description
`v1:Payment/` `v1:HostedDataID`	`xs:string`	Stores the Hosted Data ID for the Data Vault product
`v1:Payment/` `v1:HostedDataStoreID`	`xs:string`	Stores the Hosted Data ID for the Data Vault product in this store (only as technical user)
`v1:Payment/` `v1:DeclineHostedDataDuplicates`	`xs:string`	Declines duplicate credit card or German direct debit accounts
`v1:Payment/` `v1:SubTotal`	`xs:decimal`	Stores the Sub Total of an order. If this member is set, then also ChargeTotal has to be set.
`v1:Payment/` `v1:ValueAddedTax`	`xs:decimal`	Stores the VAT of an order. If this member is set, then also SubTotal has to be set.
`v1:Payment/` `v1:DeliveryAmount`	`xs:decimal`	Stores the delivery amount of an order. If this member is set, then also SubTotal has to be set.
`v1:Payment/` `v1:ChargeTotal`	`xs:double`	Stores the transaction amount. Make sure that the number of positions after the decimal point does not exceed 2, e.g. `3.123` would be invalid – however, `3.12`, `3.1`, and `3` are correct.
`v1:Payment/` `v1:Currency`	`xs:string`	Stores the currency as a three-digit ISO 4217 value (e. g. `978` for Euro)

10.1.8 TransactionDetails

Path/Name	XML Schema type	Description
`v1:TransactionDetails/` `v1:OrderId`	`xs:string`	Stores the order ID. This must be unique per Store ID. If no Order ID is transmitted, the Internet Payment Gateway will generate one automatically.
`v1:TransactionDetails/` `v1:Ip`	`xs:string`	Stores the customer's IP address which can be used by the Web Service API for fraud detection by IP address. Make sure that

		you supply the IP in the format *xxx.xxx.xxx.xxx*, e.g. `128.0.10.2` would be a valid IP.
`v1:TransactionDetails/ v1:ReferenceNumber`	`xs:string`	Stores the six digit reference number you have received as the result of a successful external authorization (e.g. by phone). The Internet Payment Gateway needs this number for uniquely mapping a *ForceTicket* transaction to a previously performed external authorization.
`v1:TransactionDetails/ v1:TDate`	`xs:string`	Stores the TDate of the *Sale*, *PostAuth*, *ForceTicket*, *Return*, or *Credit* transaction this *Void* transaction refers to. A TDate value is returned within the response to a successful transaction of one of these five types. When performing a *Void* transaction, you have to pass the TDate in addition to the order ID for uniquely identifying the transaction to be voided. The scenario presented below gives an example.
`v1:TransactionDetails/ v1:TransactionOrigin`	`xs:string`	The source of the transaction. The possible values are `ECI` (if the order was received via email or Internet), `MOTO` (mail order / telephone order) and `RETAIL` (face to face).
`v1:TransactionDetails/ v1:InvoiceNumber`	`xs:string`	Stores the invoice number.
`v1:TransactionDetails/ v1:PONumber`	`xs:string`	Stores the purchase order number.
`v1:TransactionDetails/ v1:DynamicMerchantName`	`xs:string`	Stores a dynamic merchant name for the cardholder's statement
`v1:TransactionDetails/ v1:Comments`	`xs:string`	Stores the comments.

10.1.9 InquiryRateReference

Path/Name	XML Schema type	Description
`v1:InquiryRateReference/ v1:InquiryRateId`	`xs:long`	A reference to a rate-inquiry for transactions with Global Choice™or Dynamic Pricing.
`v1:InquiryRateReference/ v1:DccApplied`	`xs:boolean`	Specifies whether a cardholder has choosen to accept the proposed currency conversion offering when using Global Choice™.

10.1.10 Billing

Path/Name	XML Schema type	Description
v1:Billing/ v1:CustomerID	xs:string	Stores your ID for your customer.
v1:Billing/ v1:Name	xs:string	Stores the customer's name. If provided, it will appear on your transaction reports.
v1:Billing/ v1:Company	xs:string	Stores the customer's company. If provided, it will appear on your transaction reports.
v1:Billing/ v1:Address1	xs:string	Stores the first line of the customer's address. If provided, it will appear on your transaction reports.
v1:Billing/ v1:Address2	xs:string	Stores the second line of the customer's address. If provided, it will appear on your transaction reports.
v1:Billing/ v1:City	xs:string	Stores the customer's city. If provided, it will appear on your transaction reports.
v1:Billing/ v1:State	xs:string	Stores the customer's state. If provided, it will appear on your transaction reports.
v1:Billing/ v1:Zip	xs:string	Stores the customer's zip code. If provided, it will appear on your transaction reports.
v1:Billing/ v1:Country	xs:string	Stores the customer's country. If provided, it will appear on your transaction reports.
v1:Billing/ v1:Phone	xs:string	Stores the customer's phone number. If provided, it will appear on your transaction reports.
v1:Billing/ v1:Fax	xs:string	Stores the customer's fax number. If provided, it will appear on your transaction reports.
v1:Billing/ v1:Email	xs:string	Stores the customer's Email address. If provided, it will appear on your transaction reports. If you are using the email transaction notification feature, this email address will be used for notifications to your customer.

10.1.11 Shipping

Path/Name	XML Schema type	Description
v1:Shipping/ v1:Type	xs:string	Stores the way of delivery.
v1:Shipping/	xs:string	Stores the name of the recipient. If provided, it will appear on your transaction

v1:Name		reports.
v1:Shipping/ v1:Address1	xs:string	Stores the first line of the shipping address. If provided, it will appear on your transaction reports.
v1:Shipping/ v1:Address2	xs:string	Stores the second line of the shipping address. If provided, it will appear on your transaction reports.
v1:Shipping/ v1:City	xs:string	Stores the recipient's city. If provided, it will appear on your transaction reports.
v1:Shipping/ v1:State	xs:string	Stores the recipient's state. If provided, it will appear on your transaction reports.
v1:Shipping/ v1:Zip	xs:string	Stores the recipient's zip code. If provided, it will appear on your transaction reports.
v1:Shipping/ v1:Country	xs:string	Stores the recipient's country. If provided, it will appear on your transaction reports.

10.1.12 TopUpTxType

Path/Name	XML Schema type	Description
v1:TopUpTxType/ v1:MPCharge/ v1:MNSP	xs:string	Stores the Mobile Network Service Provider. Possible values are : D1 (T-Mobile), D2 (Vodafone), EP (E-Plus), VI (O2).
v1:TopUpTxType/ v1:MPCharge/ v1:MSISDN	xs:string	Stores the mobile phone number that shall be used for the top-up
v1:TopUpTxType/ v1:MPCharge/ v1:PaymentType	xs:string	Stores the payment type (separate transaction). Possible values are: Amex, Cash, Diners, ECMC (MasterCard), JCB, VISA.

10.1.13 ClientLocale

Path/Name	XML Schema type	Description
v1:ClientLocale/ v1:Language	xs:string	If you are using the email transaction notification feature, this language will be used for notifications to your customer. Possible values are: de, en, it.
v1:ClientLocale/ v1:Country	xs:string	Specifies the variant of the language. This member can only be set if the language is set. Possible values are: DE, GB, IT. If you do not define a country, a matching

		country will be chosen.

11 SOAP-Antwortnachricht lesen

Die SOAP-Antwortnachricht kann man als das Resultat der Web Service Operation betrachten. Die Verarbeitung der SOAP-Anfragenachricht kann darum entweder im Erfolgsfall zu einer SOAP-Antwortnachricht (d. h. dem zurückgelieferten Parameter) führen oder im Fehlerfall (d. h. bei Auslösen einer Ausnahme) zu einer SOAP-Fehlermeldung. Beide SOAP-Nachrichtentypen sind im Body der HTTP-Antwortnachricht enthalten.

11.1 SOAP-Antwortnachricht

Eine SOAP-Antwortnachricht wird als Resultat empfangen, wenn der Kreditkartenprozessor (vom First Data Austria Internet Payment Gateway angefragt) Ihre Transaktion genehmigt hat. Die Mitteilung folgt dem Schema:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
                  <!-- transaction result -->
            </ipgapi:IPGApiOrderResponse>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Haben Sie eine Action geschickt, erhalten Sie eine `ipgapi:IPGApiActionResponse`.

Wieder sind hier keine Header definiert. Der SOAP-Body enthält das eigentliche Transaktionsresultat im Element `ipgapi:IPGApiOrderResponse` oder `ipgapi:IPGApiOrderRequest`. Seine Unterelemente und deren Bedeutung sind im nächsten Kapitel beschrieben.

Ein kurzes Beispiel: eine genehmigte *Sale* Transaktion wird in einer SOAP-Nachricht ähnlich der nachfolgenden verpackt:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
                  <ipgapi:CommercialServiceProvider>
                        BNLP
                  </ipgapi:CommercialServiceProvider>
                  <ipgapi:TransactionTime>
                        1192111687392
                  </ipgapi:TransactionTime>
                  <ipgapi:ProcessorReferenceNumber>
                        3105
                  </ipgapi:ProcessorReferenceNumber>
                  <ipgapi:ProcessorResponseMessage>
                        Function performed error-free
                  </ipgapi:ProcessorResponseMessage>
                  <ipgapi:ErrorMessage />
                  <ipgapi:OrderId>

                        62e3b5df-2911-4e89-8356-1e49302b1807

                  </ipgapi:OrderId>

                  <ipgapi:OrderId>
                        62e3b5df-2911-4e89-8356-1e49302b1807
                  </ipgapi:OrderId>
```

```
                        <ipgapi:ApprovalCode>
                              Y:440368:0000057177:PPXM:0043364291
                        </ipgapi:ApprovalCode>
                        <ipgapi:AVSResponse>PPX</ipgapi:AVSResponse>
                        <ipgapi:TDate>1192140473</ipgapi:TDate>
                        <ipgapi:TransactionResult>
                              APPROVED
                        </ipgapi:TransactionResult>
                        <ipgapi:TerminalID>123456</ipgapi:TerminalID>
                        <ipgapi:ProcessorResponseCode>
                              00
                        </ipgapi:ProcessorResponseCode>
                        <ipgapi:ProcessorApprovalCode>
                              440368
                        </ipgapi:ProcessorApprovalCode>
                        <ipgapi:ProcessorReceiptNumber>
                              4291
                        </ipgapi:ProcessorReceiptNumber>
                        <ipgapi:ProcessorTraceNumber>
                              004336
                        </ipgapi:ProcessorTraceNumber>
                  </ipgapi:IPGApiOrderResponse>
            </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

11.2 SOAP-Fehlernachricht

Im Allgemeinen hat eine vom First Data Austria API Web Service zurückgelieferte SOAP-Fehlernachricht das folgende Format:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
                  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
            <SOAP-ENV:Header />
            <SOAP-ENV:Body>
                  <SOAP-ENV:Fault>
                        <faultcode>SOAP-ENV:Client</faultcode>
                        <faultstring xml:lang="en-US">
                              <!-- fault message -->
                        </faultstring>
                        <detail>
                              <!-- fault message -->
                        </detail>
                  </SOAP-ENV:Fault>
            </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Grundsätzlich enthält das Element `faultstring` den Fehlertyp. Je nach Fehlertyp werden die anderen Elemente besetzt. Bitte beachten Sie, dass nicht alle oben gezeigten Unterelemente innerhalb des `SOAP-ENV:Fault` Elements auftreten müssen. Welche Elemente für welchen Fehlertyp existieren, ist weiter unten beschrieben.

11.2.1 SOAP-ENV:Server

Dieser Fehlertyp weist darauf hin, dass der Web Service Ihre Transaktion aufgrund eines internen Systemfehlers nicht verarbeiten konnte. Bitte kontaktieren Sie in diesem Fall unser technisches Support-Team.

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
                  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
```

```
        <SOAP-ENV:Header />
        <SOAP-ENV:Body>
                <SOAP-ENV:Fault>
                        <faultcode>SOAP-ENV:Server</faultcode>
                        <faultstring xml:lang="en-US">
                                unexpected error
                        </faultstring>
                </SOAP-ENV:Fault>
        </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Die Unterelemente der SOAP-Fehlermitteilung relativ zum Element `SOAP-ENV:Envelope/SOAP-ENV:Body/SOAP-ENV:Fault` werden wie folgt gesetzt:

Pfad / Name	XML-Schematyp	Beschreibung
`Faultcode`	`xs:string`	Dieses Element wird immer auf `SOAP-ENV:Server` gesetzt und zeigt an, dass ein Gateway-seitiger Fehler besteht.
`Faultstring`	`xs:string`	Dieses Element enthält immer den folgenden Fehlerstring: `unexpected error`

11.2.2 SOAP-ENV:Client

11.2.2.1 MerchantException

Dieser Fehlertyp tritt auf, wenn First Data Austria API den Fehler darauf zurückführen kann, dass vom Ihrem Online-Shop unkorrekte Informationen übermittelt wurden. Dies kann folgende Gründe haben:

1. Ihr Shop ist nicht freigeschaltet. Sollten Sie diese Fehlermitteilung erhalten, obwohl Ihr Shop bereits freigeschaltet sein sollte, wenden Sie sich bitte an unser technisches Support-Team.
2. Die für die HTTPS Autorisierung übermittelte Kombination von Store-ID / User-ID ist syntaktisch unkorrekt.
3. Die XML Nachricht entspricht nicht dem Schema.

Eine *MerchantException* sieht wie folgt aus:

```
<?xml version="1.0" encoding="UTF-8"?>
<SOAP-ENV:Envelope
                xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
        <SOAP-ENV:Header />
        <SOAP-ENV:Body>
                <SOAP-ENV:Fault>
                        <faultcode>SOAP-ENV:Client</faultcode>
                        <faultstring xml:lang="en-US">
                                MerchantException
                        </faultstring>
                        <detail>
                                <!-- detailed explanation. -->
                        </detail>
```

```
            </SOAP-ENV:Fault>
      </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Die Elemente der SOAP-Fehlermitteilung relativ zum Element `SOAP-ENV:Envelope/SOAP-ENV:Body/SOAP-ENV:Fault` werden wie folgt gesetzt:

Pfad / Name	XML-Schematyp	Beschreibung
`Faultcode`	`xs:string`	Dieses Element wird immer auf `SOAP-ENV:Client` gesetzt und zeigt an, dass vom Client eingereichte, unkorrekte Informationen zum Shop/ zur Order Ursache des Fehler sind.
`Faultstring`	`xs:string`	Dieses Element enthält immer den Fehlerstring `MerchantException`
`Detail/reason`	`xs:string`	Dieses Element enthält nähere Angaben zur Fehlerursache

Detaillierte Informationen zu einigen Fehlerursachen finden Sie im Abschnitt Merchant Exceptions im Anhang.

11.2.2.2 ProcessingException

Dieser Fehlertyp wird immer dann ausgegeben, wenn First Data Austria API bei der Verarbeitung Ihrer Transaktion einen Fehler erkannt hat. Der Unterschied zu den übrigen Fehlertypen besteht darin, dass First Data Austria API die Transaktion akzeptiert und an den Kreditkartenprozessor weitergeleitet hat. Erst der Prozessor hat aufgrund eines Verarbeitungsfehlers eine Ausnahme ausgelöst.

Dies ist beispielsweise der Fall, wenn das Konto des Kunden nicht genug Deckung aufweist oder die verwendete Kreditkarte abgelaufen ist. Letzterer Grund kann die folgende beispielhafte *ProcessingException* nach sich ziehen:

```
<SOAP-ENV:Envelope
            xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
      <SOAP-ENV:Header />
      <SOAP-ENV:Body>
            <SOAP-ENV:Fault>
                  <faultcode>SOAP-ENV:Client</faultcode>
                  <faultstring xml:lang="en-US">
                        ProcessingException: Processing the request
                        resulted in an error - see SOAP details for more
                        information
                  </faultstring>
                  <detail>
                        <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="https://ipg-online.com/ipgapi/schemes/ipgapi">
                              <ipgapi:CommercialServiceProvider>
                                    BNLP
                              </ipgapi:CommercialServiceProvider>
                              <ipgapi:TransactionTime>
                                    1192111156423
                              </ipgapi:TransactionTime>
                              <ipgapi:ProcessorReferenceNumber />
                              <ipgapi:ProcessorResponseMessage>
```

```
                    Card expiry date exceeded
                </ipgapi:ProcessorResponseMessage>
                <ipgapi:ErrorMessage>
                    SGS-000033: Card expiry date exceeded
                </ipgapi:ErrorMessage>
                <ipgapi:OrderId>
                    62e3b5df-2911-4e89-8356-1e49302b1807
                </ipgapi:OrderId>
                <ipgapi:ApprovalCode />
                <ipgapi:AVSResponse />
                <ipgapi:TDate>1192139943</ipgapi:TDate>
                <ipgapi:TransactionResult>
                    FAILED
                </ipgapi:TransactionResult>
                <ipgapi:TerminalID>123456</ipgapi:TerminalID>
                <ipgapi:ProcessorResponseCode/>
                <ipgapi:ProcessorApprovalCode />
                <ipgapi:ProcessorReceiptNumber />
                <ipgapi:ProcessorTraceNumber />
            </ipgapi:IPGApiOrderResponse>
        </detail>
    </SOAP-ENV:Fault>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Die Unterelemente der SOAP Fehlermitteilung relativ zum Element

`SOAP-ENV:Envelope/SOAP-ENV:Body/SOAP-ENV:Fault` werden wie folgt gesetzt:

Pfad / Name	XML-Schematyp	Beschreibung
`faultcode`	`xs:string`	Dieses Element wird immer auf `SOAP-ENV:Client` gesetzt und zeigt an, dass die Fehlerursache wahrscheinlich in ungültigen übermittelten Transaktionsdaten zu finden ist.
`faultstring`	`xs:string`	Dieses Element enthält immer den Fehlerstring `ProcessingException`
`detail/` `ipgapi:IPGApiOrderResponse`	Zusammengesetztes Element	Dieses Element enthält die vom Kreditkartenprozessor zurückgelieferte Fehlerbeschreibung. Da es viele Gründe für das Auslösen einer solchen Ausnahme gibt, enthält das folgende Kapitel eine Übersicht zu den in diesem Element enthaltenen Daten.

Detaillierte Informationen zu einigen Fehlerursachen finden Sie im Abschnitt Processing Exceptions im Anhang.

12 Transaktionsergebnis auswerten

12.1 Erfolgreiche Transaktionen

Der Statusbericht zu einer Transaktion, wie er vom Internet Payment Gateway erstellt wird, ist im Element `ipgapi:IPGApiOrderResponse` enthalten und kann als die Daten verstanden werden, die von der Web Service Operation zurückgeliefert werden. Nachfolgend sind seine Elemente relativ zum übergeordneten Element `ipgapi:IPGApiOrderResponse` beschrieben. Bitte beachten Sie, dass die Antwort immer den vollen Bestand dieser Elemente enthält, auch wenn einige von ihnen eventuell leer sind.

Pfad / Name	XML-Schematyp	Beschreibung
`ipgapi:CommercialServiceProvider`	`xs:string`	Gibt Ihren Provider an.
`ipgapi:TransactionTime`	`xs:string`	Zeitstempel des Internet Payment Gateways vor Rücklieferung der Transaktionsgenehmigung.
`ipgapi:ProcessorReferenceNumber`	`xs:string`	In manchen Fällen kann dieses Element leer sein. Es nimmt eine Nummer auf, die der Kreditkartenprozessor als Referenz auf diese Transaktion nutzen kann. Diese Nummer müssen Sie in keiner weiteren Transaktion mehr angeben. Sie sollten diese Angabe jedoch bereithalten, wenn Sie Probleme mit Ihrer Transaktion feststellen und Kontakt mit dem Support aufnehmen möchten.
`ipgapi:ProcessorResponseMessage`	`xs:string`	Im Falle einer Genehmigung enthält dieses Element folgenden String: `Function performed error-free`
`ipgapi:ProcessorResponseCode`	`xs:string`	Der Response-Code der Zahlungsverarbeitung
`ipgapi:ProcessorApprovalCode`	`xs:string`	Der Genehmigungs-Code der Zahlungsverarbeitung
`ipgapi:ProcessorReceiptNumber`	`xs:string`	Die Belegnummer der Zahlungsverarbeitung
`ipgapi:ProcessorTraceNumber`	`xs:string`	Die Trace-Nummer der Zahlungsverarbeitung
`ipgapi:ErrorMessage`	`xs:string`	Im Falle einer Genehmigung ist dieses Element leer.
`ipgapi:OrderId`	`xs:string`	Dieses Element enthält die Order-ID. Für *Sale*, *PreAuth*, *ForceTicket* und *Credit* Transaktionen wird eine neue Order-ID zurückgeliefert. Bei

		PostAuth, *Return* und *Void* Transaktionen tragen Sie diese Nummer in das Element `v1:OrderId` ein, um klarzustellen, auf welche Transaktion Sie sich beziehen. Das Element `ipgapi:OrderId` einer Genehmigung für eine *PostAuth*, *Return* oder *Void* Transaktion liefert einfach die Order-ID zurück, auf die sich die Transaktion bezogen hat.
`ipgapi:ApprovalCode`	`xs:string`	Speichert den Genehmigungscode, den der Transaktionsprozessor für diese Transaktion generiert hat. Diesen Code müssen Sie in keiner weiteren Transaktion mehr angeben. Sie sollten diese Angabe jedoch bereithalten, wenn Sie Probleme mit Ihrer Transaktion feststellen und Kontakt mit dem Support aufnehmen möchten.
`ipgapi:AVSResponse`	`xs:string`	Enthält die Antwort des Adressenverifikationssystems (AVS).
`ipgapi:TDate`	`xs:string`	Speichert das TDate, das Sie angeben müssen, um diese Transaktion unwirksam zu machen (was nur bei *Sale* und *PostAuth* Transaktionen möglich ist). In diesem Fall übertragen Sie den Wert in das Element `v1:TDate` der *Void* Transaktion, die Sie erstellen möchten.
`ipgapi:TransactionResult`	`xs:string`	Speichert das Transaktionsresultat, das im Falle einer Genehmigung immer `APPROVED` lautet.
`ipgapi:TerminalID`	`xs:string`	Die bei der Transaktion verwendete Terminal ID.
`ipgapi:PaymentType`	`xs:string`	Der Payment-Typ, der für die Transaktion genutzt wird.
`ipgapi:Brand`	`xs:string`	Der Name der Kreditkarte, die für die Transaktion genutzt wurde.
`ipgapi:Country`	`xs:string`	Das kartenherausgebende Land.

12.2 Fehlerhafte Transaktionen

Eine SOAP-Fehlernachricht enthält ein Element `ipgapi:IPGApiOrderResponse`, das als Unterelement eines SOAP-Elements `detail` übergeben wird. Bitte beachten Sie, dass dessen Unterelemente exakt die gleichen wie im Falle einer Transaktionsgenehmigung sind. Ihre Bedeutung im Falle eines Fehlers ist nachfolgend erklärt:

Pfad / Name	XML-Schematyp	Beschreibung
`ipgapi: CommercialServiceProvider`	`xs:string`	Gibt Ihren Provider an.
`ipgapi:TransactionTime`	`xs:string`	Zeitstempel des Internet Payment Gateways vor Rücklieferung des Transaktionsfehlers.
`ipgapi: ProcessorReferenceNumber`	`xs:string`	In manchen Fällen kann dieses Element leer sein. Nimmt eine Nummer auf, die der Kreditkartenprozessor als Referenz auf diese Transaktion nutzen kann. Diese Nummer müssen Sie in keiner weiteren Transaktion mehr angeben. Sie sollten diese Angabe jedoch bereithalten, wenn Sie Probleme mit Ihrer Transaktion feststellen und Kontakt mit dem Support aufnehmen möchten.
`ipgapi: ProcessorResponseMessage`	`xs:string`	Speichert die vom Kreditkartenprozessor zurückgelieferte Fehlermeldung. Im Falle einer abgelaufenen Kreditkarte wäre dies beispielsweise: `Card expiry date exceeded`
`ipgapi:ErrorMessage`	`xs:string`	Speichert die vom Internet Payment Gateway zurückgelieferte Fehlermeldung. Sie folgt immer der Syntax *SGS-XXXXXX: Message*, wobei *XXXXXX* ein sechsstelliger Fehlercode ist und *Message* den Fehler beschreibt (diese Beschreibung kann von der Mitteilung des Prozessors abweichen). Im Beispiel oben könnte die Fehlermeldung `SGS-000033: Card expiry date exceeded` zurückgeliefert werden. Halten Sie den Fehlercode und die Mitteilung bereit, wenn Sie sich an den Support wenden.
`ipgapi:OrderId`	`xs:string`	Speichert die Order-ID. Im Gegensatz zu einer Genehmigung wird diese Order-ID niemals für eine weitere Transaktion benötigt, jedoch für die Ermittlung der Fehlerursache. Halten Sie darum diese Angabe bereit, wenn Sie sich an den Support wenden.
`ipgapi:ApprovalCode`	`xs:string`	Im Falle eines Transaktionsfehlers ist dieses Element leer.

ipgapi:AVSResponse	xs:string	Enthält die Antwort des Adressenverifikationssystems (AVS).
ipgapi:TDate	xs:string	Speichert das TDate. Ähnlich wie die Order-ID wird das TDate niemals für eine weitere Transaktion benötigt, jedoch für die Ermittlung der Fehlerursache. Halten Sie darum diese Angabe bereit, wenn Sie sich an den Support wenden.
ipgapi:TransactionResult	xs:string	Im Fehlerfall kann es drei mögliche Werte geben: • DECLINE • FRAUD • FAILED DECLINE wird zurückgeliefert, wenn der Kreditkartenprozessor die Transaktion ablehnt, z. B. weil das Konto des Kunden keine ausreichende Deckung aufweist. FRAUD wird zurückgeliefert, wenn das Internet Payment Gateway einen Betrugsversuch vermutet. Sollte es bei der Verarbeitung zu einem internen Fehler des Gateways kommen, wird FAILED zurück geliefert.
ipgapi:TerminalID	xs:string	Die bei der Transaktion verwendete Terminal ID.

13 HTTPS POST-Anfrage erstellen

Eine HTTPS POST-Anfrage werden Sie nur sehr selten eigenhändig erstellen müssen. Für die Zusammenstellung von HTTPS-Anfragen gibt es eine Vielzahl von Tools und Bibliotheken, die Sie bei dieser Aufgabe unterstützen. In den meisten Fällen ist die dafür erforderliche Funktionalität Bestandteil des Standardbestands von Bibliotheken, der mit der technologischen Umgebung ausgeliefert wird, die Sie zur Entwicklung Ihres Online-Shops einsetzen.

Da es bei der Nutzung dieser Bibliotheken geringfügige Unterschiede gibt, kann an dieser Stelle keine allgemeingültige Vorgehensweise beschrieben werden. Zur Darstellung der Grundkonzepte zeigen die folgenden Kapitel Beispiele für die Erstellung einer gültigen HTTPS-Anfrage in PHP und ASP. Im Allgemeinen müssen Sie den folgenden Parameterbestand bereitstellen, um unabhängig von der verwendeten Programmiersprache eine gültige HTTP-Anfrage zu erstellen:

Parameter	Wert	Beschreibung

URL	`https://`
	`test.ipg-online.com/`
	`ipgapi/services`
		Dies ist die vollständige URL des First Data Austria API Web Service. Abhängig von der Funktionalität, die Sie für die Erstellung von HTTPS-Anfragen einsetzen, müssen Sie die URL eventuell in Host und Service aufteilen und diese Angaben in die entsprechenden Header der HTTPS-Anfrage eintragen.

Content-Type	`text/xml`	Dieser zusätzliche HTTPS-Header muss ebenfalls festgelegt werden. Grund dafür ist die SOAP-Anfragenachricht, die in XML kodiert und als Inhalt im Body der HTTPS POST-Anfrage transportiert werden soll.

Authorization	Type: `Basic`
	Username: `WS`*storeID*`._.`*userID*
	Password: *yourPassword*
		Die Identität Ihres Shops wird vom Internet Payment Gateway anhand dieser Identifizierungsangaben festgestellt. Um First Data Austria API verwenden zu können, müssen Sie Ihre Store-ID, Ihre User-ID und das Passwort als Inhalt eines HTTPS *Basic* Autorisierungsheaders übermitteln. Beispiel: Wenn Ihre Store-ID `101`, Ihre User-ID `007` und Ihr Passwort `myPW` lauten, dann ist der Benutzername für die Autorisierung `WS101._.1`. Der komplette HTTPS-Autorisierungsheader würde so aussehen:
		`Authorization: Basic`
		`V1MxMDEuXy4wMDc6bXlQVw==`
		Bitte beachten Sie, dass die letzte Zeichenfolge das Ergebnis einer Base64-Kodierung des Strings `WS101._.1:myPW` ist.

HTTP-Body	SOAP-Anfrage XML	Der Body der HTTP POST-Anfrage enthält die SOAP-Anfragemitteilung.

13.1 PHP

HTTP Kommunikation in PHP erfolgt meist mit Hilfe von cURL, das sowohl als Bibliothek als auch als Befehlszeilen-Tool erhältlich ist. In neuere PHP-Versionen ist cURL bereits als Erweiterung integriert, die noch "aktiviert" werden muss, damit die cURL-Funktionalität in jedem PHP-Skript verfügbar ist. Wenn Ihr Web-Server unter Microsoft Windows läuft, ist dies schnell erledigt. Auf Unix/Linux-Maschinen muss dazu PHP möglicherweise erst kompiliert werden. Darum könnten Sie anstelle der Verwendung der cURL-Erweiterung auch in Betracht ziehen, das cURL-Befehlszeilentool Ihres PHP-Skripts aufzurufen. Nachfolgend sind beide Vorgehensweisen berücksichtigt, wobei wir mit der Nutzung der cURL-Erweiterung in PHP 5.2.4 auf einer Windows Maschine beginnen.

13.1.1 cURL PHP-Erweiterung nutzen

In den meisten Fällen genügt für die Aktivierung der cURL-Erweiterung in PHP 5.2.4, die Kommentarkennzeichnung der folgenden Zeile in der *php.ini* Konfigurationsdatei zu entfernen:

```
;extension=php_curl.dll
```

Bitte beachten Sie, dass andere PHP-Versionen andere Vorgehensweisen erfordern könnten, um die cURL-Unterstützung in PHP zu aktivieren. Weitere Informationen hierzu finden Sie in Ihrer PHP-Dokumentation. Nach Aktivierung von cURL wird eine HTTP-Anfrage mit den oben genannten Parametern mittels folgender PHP-Statements erstellt:

```
<?php
// storing the SOAP message in a variable – note that the plain XML code
// is passed here as string for reasons of simplicity, however, it is
// certainly a good practice to build the XML e.g. with DOM – furthermore,
// when using special characters, you should make sure that the XML string
// gets UTF-8 encoded (which is not done here):
$body = "<SOAP-ENV:Envelope ...>...</SOAP-ENV:Envelope>";
// initializing cURL with the IPG API URL:
$ch = curl_init("https://test.ipg-online.com/ipgapi/services");
// setting the request type to POST:
curl_setopt($ch, CURLOPT_POST, 1);
// setting the content type:
curl_setopt($ch, CURLOPT_HTTPHEADER, array("Content-Type: text/xml"));
// setting the authorization method to BASIC:
curl_setopt($ch, CURLOPT_HTTPAUTH, CURLAUTH_BASIC);
// supplying your credentials:
curl_setopt($ch, CURLOPT_USERPWD, "WS101._.1:myPW");
// filling the request body with your SOAP message:
curl_setopt($ch, CURLOPT_POSTFIELDS, $body);
...
?>
```

Wie die für eine SSL-Kommunikation notwendigen Sicherheitsoptionen eingestellt werden, ist im nächsten Kapitel in Form einer Erweiterung des obigen Skripts beschrieben.

13.1.2 cURL-Befehlszeilentool nutzen

Aus den oben genannten Gründen ziehen Sie es vielleicht vor, anstelle der Erweiterung das cURL-Befehlszeilentool zu verwenden. Die Nutzung dieses Tools erfordert keinerlei PHP-Konfigurationsbemühungen. Ihr PHP-Skript muss nur die ausführbare Programmdatei mit einem Bestand von Parametern aufrufen. Da die Sicherheitseinstellungen erst im folgenden

Kapitel behandelt werden, zeigt das Skript unten nur die Einrichtung der HTTP-Standardparameter. Die SSL-Parameter kommen im nächsten Kapitel hinzu.

```
<?php
// storing the SOAP message in a variable – note that you have to escape
// " and \n, since the latter makes the command line tool fail,
// furthermore note that the plain XML code is passed here as string
// for reasons of simplicity, however, it is certainly a good practice
// to build the XML e.g. with DOM – finally, when using special
// characters, you should make sure that the XML string gets UTF-8 encoded
// (which is not done here):
$body = "<SOAP-ENV:Envelope ...>...</SOAP-ENV:Envelope>";
// setting the path to the cURL command line tool – adapt this path to the
// path where you have saved the cURL binaries:
$path = "C:\curl\curl.exe";
// setting the IPG API URL:
$apiUrl = " https://test.ipg-online.com/ipgapi/services";
// setting the content type:
$contentType = " --header \"Content-Type: text/xml\"";
// setting the authorization method to BASIC and supplying
// your credentials:
$user = " --basic --user WS101._.1:myPW";
// setting the request body with your SOAP message – this automatically
// marks the request as POST:
$data = " --data \"".$body."\"".
...
?>
```

13.2 ASP

Für die Erstellung einer HTTP-Anfrage in ASP gibt es mehrere Wege. Nachfolgend ist hierzu die Verwendung von WinHTTP 5.1 beschrieben, da es Bestandteil von Windows Server 2003 und Windows XP SP2 ist. Außerdem werden nur wenige Codezeilen benötigt, um eine gültige HTTP-Anfrage zu erstellen. Bitte beachten Sie, dass das nachfolgende Codefragment in JavaScript verfasst ist. Wenn Sie stattdessen VB Script verwenden, ändern sich die gezeigten Statements nicht grundlegend.

```
<%@ language="javascript"%>
<html>...<body>
<%
// storing the SOAP message in a variable – note that the plain XML code
// is passed here as string for reasons of simplicity, however, it is
// certainly a good practice to build the XML e.g. with DOM – furthermore,
// when using special characters, you should make sure that the XML string
// gets UTF-8 encoded (which is not done here):
var body = "<SOAP-ENV:Envelope ...>...</SOAP-ENV:Envelope>";
// constructing the request object:
var request = Server.createObject("WinHttp.WinHttpRequest.5.1");
// initializing the request object with the HTTP method POST
// and the IPG API URL:
request.open("POST", "https://test.ipg-online.com/ipgapi/services");
// setting the content type:
request.setRequestHeader("Content-Type", "text/xml");
// setting the credentials:
request.setCredentials("WS10036000750._.1", "testinger", 0);
...
%>
</body></html>
```

Bitte beachten Sie, dass das obige Skript im nächsten Kapitel um die Einstellung der Sicherheitsoptionen erweitert wird, die für den Aufbau eines SSL-Kanals erforderlich sind.

14 SSL-Verbindung aufbauen

Vor dem Versand der HTTP-Anfrage, deren Erstellung im vorangehenden Kapitel beschrieben wurde, muss ein sicherer Kommunikationskanal aufgebaut werden. Er garantiert, dass sämtlich übermittelten Daten verschlüsselt sind und dass Client (Ihre Applikation) wie Server (auf dem der First Data Austria API Web Service läuft) sicher sein können, miteinander und mit sonst niemandem zu kommunizieren.

Beides wird durch den Aufbau einer SSL-Verbindung zwischen Client und Server erreicht, die Zertifikate austauschen. Ein Zertifikat identifiziert eindeutig einen Kommunikationspartner. Dieser Prozess läuft grundsätzlich wie folgt ab:

1. Der Client leitet den Aufbau einer sicheren Verbindung ein, indem er sein Client-Zertifikat an den Server sendet.
2. Der Server nimmt das Client-Zertifikat entgegen und gleicht es mit dem Client-Zertifikat ab, das er für diesen Client gespeichert hat.
3. Ist es gültig, antwortet der Server, indem er sein Server-Zertifikat sendet.
4. Der Client nimmt das Server-Zertifikat entgegen und gleicht es mit dem vertrauenswürdigen Server-Zertifikat ab.
5. Ist es gültig, öffnen beide Parteien einen gemeinsamen SSL-Kanal, weil sie sicher sein können, miteinander und mit sonst niemandem zu kommunizieren. Alle zwischen den Parteien ausgetauschten Daten sind verschlüsselt.

Gemäß diesem Ablauf muss Ihre Applikation zwei Dinge tun: Erstens die Kommunikation durch Senden des Client-Zertifikats eröffnen. Zweites das empfangene Server-Zertifikat verifizieren. Wie dies geschieht, ist von Plattform zu Plattform verschieden. Um jedoch die Grundkonzepte zu erläutern, werden nachfolgend die im vorangehenden Kapitel begonnenen PHP- und ASP-Skripts um die für den Aufbau einer SSL-Verbindung benötigten Statements erweitert.

14.1 PHP

Bei der Beschreibung des Aufbaus einer sicheren HTTP-Kommunikation wird in den beiden folgenden Abschnitten die Unterscheidung zwischen PHP-cURL-Erweiterung und Befehlszeilentool fortgeführt. Unabhängig von der von Ihnen letztlich verwendeten Methode werden Sie mit einem speziellen Merkmal von cURL konfrontiert: cURL verlangt, dass das Client-Zertifikat als PEM-Datei übermittelt wird und der zugehörige private Schlüssel in einer gesonderten Datei. Abschließend muss das Passwort zum privaten Schlüssel des Client-Zertifikats gesendet werden. Einfach gesagt, enthält die PEM-Datei das Zertifikat mit allen Informationen, die der Server benötigt, um den Client zu identifizieren. Der private Schlüssel ist für diese Art der Kommunikation nicht wirklich notwendig. Er ist jedoch unentbehrlich, damit cURL funktioniert.

14.1.1 PHP-cURL-Erweiterung nutzen

Ausgangsbasis ist das im vorangehenden Kapitel begonnene Skript. Die für den Aufbau einer SSL-Verbindung mit cURL notwendigen Parameter werden in den folgenden Statements festgelegt:

```
<?php
...
// telling cURL to verify the server certificate:
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, 1);
// setting the path where cURL can find the certificate to verify the
// received server certificate against:
curl_setopt($ch, CURLOPT_CAINFO, "C:\certs\geotrust.pem");
// setting the path where cURL can find the client certificate:
```

```
curl_setopt($ch, CURLOPT_SSLCERT, "C:\certs\WS101._.1.pem");
// setting the path where cURL can find the client certificate's
// private key:
curl_setopt($ch, CURLOPT_SSLKEY, "C:\certs\WS101._.1.key");
// setting the key password:
curl_setopt($ch, CURLOPT_SSLKEYPASSWD, "ckp_1193927132");
...
?>
```

Bitte beachten Sie, dass dieses Skript im nächsten Kapitel um die Statements erweitert wird, die für die eigentliche HTTP-Anfrage notwendig sind.

14.1.2 cURL-Befehlszeilentool nutzen

Ausgehend von dem im vorangehenden Kapitel begonnenen Skript lauten die Statements zur Initialisierung der SSL-Parameter, die an das cURL-Befehlszeilentool weiterzugeben sind, wie folgt

```
<?php
...
// setting the path where cURL can find the certificate to verify the
// received server certificate against:
$serverCert = " --cacert C:\certs\geotrust.pem";
// setting the path where cURL can find the client certificate:
$clientCert = " --cert C:\certs\WS101._.1.pem";
// setting the path where cURL can find the client certificate's
// private key:
$clientKey = " --key C:\certs\WS101._.1.key";
// setting the key password:
$keyPW = " --pass ckp_1193927132";
...
?>
```

Bitte beachten Sie, dass dieses Skript im nächsten Kapitel um die Statements erweitert wird, die für die eigentliche HTTP-Anfrage notwendig sind.

14.2 ASP

Damit der oben beschriebene Initialisierungsprozess funktioniert, erwartet ASP, dass Client- und Server-Zertifikat in Zertifikate-Stores vorliegen. Mit anderen Worten müssen erst beide Zertifikate installiert werden, ehe ASP via SSL kommunizieren kann. Die folgenden Schritte leiten Sie durch diesen Einrichtungsprozess. Sie gehen davon aus, dass ASP in Microsoft IIS 5.1 unter Windows XP läuft:

1. Klicken Sie auf *Start*, dann auf *Run...*. Geben Sie *mmc* ein und klicken Sie auf *OK*.
2. Öffnen Sie das Menü *File* und wählen Sie *Add/Remove Snap-In*.
3. Klicken Sie auf *Add*.
4. Wählen Sie unter *Snap-In* die Option *Certificates* und klicken Sie auf *Add*.
5. Sie werden aufgefordert, das Konto auszuwählen, für dass Sie die Zertifikate verwalten wollen. Da IIS den Computer-Account nutzt, wählen Sie *Computer Account* und klicken auf *Next*.
6. Wählen Sie *Local Computer* und klicken Sie auf *Finish*.
7. Klicken Sie auf *Close* und dann auf *OK*.
8. Entfalten Sie den Zweig *Certificates (Local Computer)*. Das Client-Zertifikat wird im Ordner *Personal* installiert.
9. Klicken Sie mit der rechten Maustaste darum auf den Ordner *Certificates*, wählen Sie *All Tasks* und klicken Sie auf *Import...* Damit rufen Sie den Certificate Import Wizard auf.

10. Klicken Sie auf *Next*. Wählen Sie die p12-Datei mit Ihrem Client-Zertifikat und klicken Sie auf *Next*. Bitte beachten Sie, dass dieser Prozess genau so abläuft, wie in Kapitel 3 beschrieben.
11. Geben Sie das Passwort für die Installation des Client-Zertifikats ein und klicken Sie auf *Next*.
12. Wählen Sie *Place all certificates in the following store* und suchen Sie den Ordner *Personal*, sofern er noch nicht angezeigt wird. Klicken Sie auf *Next*.
13. Prüfen Sie die angezeigten Einstellungen und klicken Sie auf *Finish*. Ihr Client-Zertifikat ist damit im Zertifikate-Store des lokalen Computers installiert. Hier kann IIS (und das darin laufende ASP) bei der HTTP-Kommunikation mit einem anderen Server das Client-Zertifikat zur Prüfung finden.
14. Als nächstes muss das Server-Zertifikat im Store *Trusted Root Certification Authorities* installiert werden. Die Zertifikate in diesem Store werden zur Verifikation genutzt, wann immer ein Zertifikat von einem Server eintrifft. Darum muss das Server-Zertifikat von First Data Austria API ebenfalls hier installiert werden. Auf diese Weise kann IIS das bei der Kontaktaufnahme mit dem Web Service empfangene Server-Zertifikat auf seine Vertrauenswürdigkeit prüfen. Wählen Sie *Trusted Root Certification Authorities* aus dem Zweig *Certificates (Local Computer)* und öffnen Sie den Ordner *Certificates*.
15. Klicken Sie mit der rechten Maustaste auf den Ordner *Certificates*, wählen Sie *All Tasks* und klicken Sie auf *Import...* Damit rufen Sie den Certificate Import Wizard auf.
16. Klicken Sie auf *Next*. Wählen Sie die PEM-Datei mit dem Server-Zertifikat und klicken Sie auf *Next*. Bitte beachten Sie, dass dieser Prozess genau so abläuft, wie in Kapitel 3 beschrieben.
17. Wählen Sie *Place all certificates in the following store* und suchen Sie den Ordner *Trusted Root Certification Authorities*, sofern er noch nicht angezeigt wird. Klicken Sie auf *Next*.
18. Prüfen Sie die angezeigten Einstellungen und klicken Sie auf *Finish*. Das Server-Zertifikat ist damit im Store des lokalen Computers für vertrauenswürdige Zertifikate installiert. Hier kann IIS das Server-Zertifikat zum Abgleich mit dem First Data Austria API Server-Zertifikat heranziehen, das während des SSL-Einrichtungsprozesses empfangen wurde.

Nach der Installation beider Zertifikate sollte man davon ausgehen können, dass die Umgebung, die ASP eine Kommunikation via SSL erlaubt, fertig eingerichtet ist. Eine Sache bleibt jedoch noch zu tun, damit die Kommunikation nicht fehlschlägt: IIS, unter dem Ihr ASP läuft, hat einen Windows Anwender, der nicht über die nötigen Rechte für den Zugriff auf den private Schlüssel des Client-Zertifikats verfügt. Auch wenn der Zugriff auf den privaten Schlüssel nicht wirklich notwendig für den Aufbau einer SSL-Verbindung mit dem Internet Payment Gateway ist, braucht der IIS-Anwender Zugriffsrechte für die Ausführung des Authentifizierungsprozesses in ASP. Für die Zuweisung von Rechten an einen Anwender hält Microsoft das Tool *WinHttpCertCfg.exe* zum kostenlosen Download bereit. Sie finden es unter:

```
http://www.microsoft.com/downloads/details.aspx?familyid=c42e27ac-3409-
40e9-8667-c748e422833f&displaylang=en
```

Nach Installation des Tools öffnen Sie ein Befehlseingabefenster, wechseln in das Verzeichnis, in dem Sie das Tool installiert haben, und geben die folgende Befehlszeile ein, um dem IIS Anwender die Zugriffsrechte zuzuteilen:

```
winhttpcertcfg -g -c LOCAL_MACHINE\My -s WS101._.1 -a IWAM_MyMachine
```

`LOCAL_MACHINE\My` gibt den Schlüssel-Store an, in dem die persönlichen Zertifikate für den lokalen Maschinen-Account gespeichert sind. Nach Installation des Client-Zertifikats im persönlichen Zertifikate-Store wie oben beschrieben, ist das Client-Zertifikat unter dem genannten Pfad zu finden. Es muss dann kein anderer Pfad angegeben werden. `WS101._.1` ist der Name des Client-Zertifikats. Diesen Namen müssen Sie durch den Namen Ihres Client-Zertifikats ersetzen. Dazu schauen Sie nach, welcher Name für das Client-Zertifikat in der *mmc* Konsole angezeigt wird, nachdem Sie es wie oben beschrieben installiert haben. `IWAM_MyMachine` schließlich gibt den IIS Benutzernamen an. Beachten Sie, dass IIS 5.1 in der Voreinstellung `IWAM_`*MachineName* verwendet. Das bedeutet, wenn Ihre Maschine den Namen *IISServerMachine* trägt, lautet der IIS-Benutzername `IWAM_IISServerMachine`. Bitte beachten Sie, dass andere IIS-Versionen eventuell einem anderen Namensschema folgen. Wenn Sie den Namen Ihrer Maschine oder den IIS-Benutzernamen nicht kennen, schlagen Sie in der IIS-Dokumentation nach oder wenden Sie sich an Ihren Administrator.

Nun können Sie SSL in Ihrem ASP-Code nutzen. Der gegenüber dem im vorangehenden Kapitel begonnenen Skript hinzugekommene Code beschränkt sich auf nur ein zusätzliches Statement. Es teilt WinHTTP mit, welches Client-Zertifikat gesendet werden soll (und wo es zu finden ist), wenn es zur Kontaktaufnahme mit dem Internet Payment Gateway kommt:

```
<%@ language="javascript"%>
<html>...<body>
<%
...
// setting the path where the client certificate to send can be found:
request.setClientCertificate("LOCAL_MACHINE\\My\\WS101._.1");
...
%>
</body></html>
```

Beachten Sie, das der Code bei Verwendung von VB Script weitgehend gleich aussieht. Vergessen Sie jedoch nicht, die doppelten Backslashs im Pfad durch einzelne zu ersetzen (d. h. der Pfadname des Zertifikats würde dann wie folgt lauten: `"LOCAL_MACHINE\My\WS101._.1"`).

Bitte beachten Sie, dass dieses Skript im nächsten Kapitel um die Statements erweitert wird, die für die eigentliche HTTP-Anfrage notwendig sind.

15 HTTPS POST-Anfrage senden und die Antwort empfangen

Die eigentliche Kommunikation mit dem First Data Austria API Web Service beginnt, wenn die HTTPS-Anfrage gesendet und auf eine Antwort gewartet wird. Wie dies geschieht, ist wiederum von der Programmiersprache abhängig, die Sie verwenden. Die meisten HTTP-Bibliotheken decken die zugrunde liegenden Kommunikationsdetails voll ab und verkürzen diesen Prozess auf einen einzigen Operationsaufruf, der als Ergebnisobjekt die HTTP-Antwort zurückliefert.

Die erforderlichen Parameter, um erfolgreich eine HTTP POST-Anfrage über SSL durchzuführen und eine Antwort (mit einem `200` HTTP-Statuscode als Inhalt) zu empfangen, sind in den beiden vorangehenden Kapiteln beschrieben. Die Festlegung ungültiger oder falscher Parameter führt dazu, dass der Webserver, auf dem der First Data Austria API Web Service läuft, einen HTTP-Standardfehlercode im HTTP-Header der Antwort zurückgibt oder einen SSL-Fehler sendet. Die Bedeutung dieser Codes können Sie jedem HTTP/SSL-Leitfaden entnehmen.

Es gibt jedoch eine wichtige Ausnahme: Wenn die von Ihnen angegebenen HTTP-Parameter korrekt sind, der Web Service aber aufgrund eines unkorrekten Wertes in der SOAP-Anfragenachricht (z. B. eine ungültige Kreditkartennummer) Ihre Transaktion nicht verarbeiten konnte, wird eine SOAP-Ausnahme ausgelöst und in den Body einer HTTP-Antwort eingetragen, die den Fehlercode `500` enthält. Einzelheiten zur Ursache der Ausnahme enthält die SOAP-Fehlermitteilung, die im Kontext des folgenden Kapitels beschrieben wird.

Zum Abschluss der in den vorangehenden Kapiteln schrittweise aufgebauten PHP- und ASP-Skripts steuern die beiden nächsten Kapitel die notwendigen Statements für die Durchführung eines HTTP-Aufrufs mittels dieser Programmiersprachen bei.

15.1 PHP

Auch hier erfolgt wieder eine zweigeteilte Erklärung für die PHP-cURL-Erweiterung bzw. die Nutzung des cURL-Befehlszeilentools:

15.1.1 PHP-cURL-Erweiterung nutzen

Das PHP-Skript unter Nutzung der cURL-Erweiterung wird vervollständigt durch die nachfolgenden Statements zur Ausführung des HTTP-Aufrufs. Beachten Sie, dass der HTTP-Aufruf eine SOAP-Antwortmitteilung oder Fehlermeldung im Body der HTTP-Antwort zurückliefert.

```
<?php
...
// telling cURL to return the HTTP response body as operation result
// value when calling curl_exec:
curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1);
// calling cURL and saving the SOAP response message in a variable which
// contains a string like "<SOAP-ENV:Envelope ...>...</SOAP-ENV:Envelope>":
$result = curl_exec($ch);
// closing cURL:
curl_close($ch);
?>
```

15.1.2 cURL-Befehlszeilentool nutzen

Die Ausführung des HTTP-Aufrufs mit dem cURL-Befehlszeilentool besteht einfach darin, das Kommando in der Befehlszeile zu vervollständigen und das externe Tool auszuführen. Das Lesen der HTTP-Antwort jedoch ist komplizierter, da das PHP-Kommando `exec` jede von einem externen Programm zurückgelieferte Zeile als Element eines Arrays speichert.

Erst die Verknüpfung aller Elemente des Arrays ergibt die SOAP-Antwort oder Fehlermeldung, die im Body der HTTP-Antwort zurückgeliefert wurde. Die folgenden Statements handhaben den HTTP-Aufruf und schließen das Skript ab:

```
<?php
...
// saving the whole command in one variable:
$curl = $path.
       $data.
       $contentType.
       $user.
       $serverCert.
       $clientCert.
       $clientKey.
       $keyPW.
       $apiUrl;
// preparing the array containing the lines returned by the cURL
// command line tool:
$returnArray = array();
// performing the HTTP call by executing the cURL command line tool:
exec($curl, $returnArray);
// preparing a variable taking the complete result:
$result = "";
// concatenating the different lines returned by the cURL command
// line tool - this result in the variable $result carrying the entire
// SOAP response message as string:
foreach($returnArray as $item)
       $result = $result.$item;
?>
```

15.2 ASP

Die Durchführung des eigentliche HTTP-Aufrufs mit WinHTTP in ASP besteht aus einem einfachen Operationsaufruf mit der SOAP-Anfrage-XML als Parameter. Nach erfolgreicher Durchführung der Anfrage wird eine SOAP-Antwort oder Fehlermeldung zurückgeliefert, die durch Zugriff auf die `responseText` Eigenschaft des Anfrageobjekts als Zeichenkette ausgelesen werden kann. Wie eine solche SOAP-Antwortnachricht aussieht, ist im nächsten Kapitel beschrieben. Die folgenden Statements vervollständigen das ASP-Skript:

```
<%@ language="javascript"%>
<html>...<body>
<%
...
// doing the HTTP call with the SOAP request message as input:
request.send(body);
// saving the SOAP response message in a string variable:
var response = request.responseText;
%>
</body></html>
```

16 Verwendung eines Java Client zur Nutzung des Web Service

First Data Austria bietet einen Java Client an, der Ihnen die Nutzung des Web Service des Internet Payment Gateways schnell und einfach ermöglicht. Eine Instanz der IPGApiClient Klasse steuert die Verbindung mit dem Web Service, erstellt XML und SOAP Nachrichten und wertet die Antwortnachrichten aus. Um eine Transaktion zu generieren oder eine Antwort zu verarbeiten, werden einfache Java Bean Klassen verwendet.

16.1 Erzeugung einer IPGApiClient Instanz

Der IPGApiClient kann mit verschiedenen Konstruktoren instantiiert werden. Das folgende Beispiel zeigt wie der einfachste Konstruktor verwendet wird. Zur Vollständigkeit und Vereinfachung des Beispiels ist auch die getBytes Methode angegeben.

```
String url = "https://test.ipg-online.com/ipgapi/services";
String storeId = "your store id";
String password = "your password";
byte[] key = getBytes("/path/to/your/keyStore.ks");
String keyPW = "your key store password";

IPGApiClient client = new IPGApiClient(url, storeId, password, key, keyPW);

/**
 * getBytes
 * reads a resource and returns a byte array
 * @param resource the resource to read
 * @return the resource as byte array
 */
public static byte[] getBytes(final String resource) throws IOException {
	final InputStream input = IO.class.getResourceAsStream(resource);
	if (input == null) {
		throw new IOException(resource);
	}
	try {
		final byte[] bytes = new byte[input.available()];
		input.read(bytes);
		return bytes;
	} finally {
		try {
			input.close();
		} catch (IOException e) {
			log.warn(resource);
		}
	}
}
```

16.2 Transaktionen erstellen und Antwort verarbeiten

Es gibt verschiedene Klassen für Kreditkarten- und Lastschrifttransaktionen. Für die Erstellung der benötigten Klassen, kann die folgende Factory Class verwendet werden:

```
de.firstdata.ipgapi.client.transaction.IPGApiTransactionFactory
```

Das folgende Beispiel zeigt eine ‘Sale’ Transaktion mit Kreditkarte und einem Betrag von 7 Euro:

```
Amount amount = new Amount("7", "978"); // ISO 4217: EUR = 978
CreditCard cC = new CreditCard("1111222233334444", "07", "17", null);
CCSaleTransaction transaction =
      IPGApiTransactionFactory.createSaleTransactionCredit(amount, cC);
// some transactions may include further information e.g. the customer
transaction.setName("a name");
try {
      IPGApiResult result = client.commitTransaction(transaction);
      // now you can read the conclusion
      System.out.println(result.getOrderId());
      System.out.println(result.getTransactionTime());
      // ...
} catch (ProcessingException e) {
      // ERROR: transaction not passed
}
```

16.3 Erstellen einer Aktion

Die folgende Factory Class kann zur Erstellung der benötigten Klasse verwendet werden:

```
de.firstdata.ipgapi.client.transaction.IPGApiActionFactory
```

Um eine Aktion auszuführen muss die *commitAction* Methode des IPGApiClient verwendet werden. Die weitere Vorgehensweise erfolgt entsprichend der Zahlungstransaktionen.

16.4 Einsatz eines Proxys

Bevor der IPGApiClient hinter einem Proxy verwendet werden kann, muss die Proxy Konfiguration des Clients mit der IPGApiClient Methode vorgenommen werden:

```
IPGApiClient.setProxy(
      final String host, final Integer port,
      final String user, final String domain, final String password)
```

Die Parameter `user`, `domain` und `password` sollten leer sein, sofern keine Identifikation benötigt wird. Wenn eine Identifikation auf einem MS Windows Proxy benötigt wird, muss der Parameter `domain` gesetzt werden. Bei Systemen wie Unix muss der Parameter `domain` leer sein.

Weiterführende Informationen finden Sie unter den folgenden Links:

http://hc.apache.org/httpclient-3.x/apidocs/org/apache/commons/httpclient/UsernamePasswordCredentials.html#UsernamePasswordCredentials(java.lang.String,%20java.lang.String)

http://hc.apache.org/httpclient-3.x/apidocs/org/apache/commons/httpclient/NTCredentials.html#NTCredentials(java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String)

Nachdem sie Proxy Parameter gesetzt sind, muss die IPGApiClient.init() Methode aufgerufen werden.

17 Anhang

ipgapi.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi" xmlns:a1="http://ipg-online.com/ipgapi/schemas/a1"
   xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1" elementFormDefault="qualified" targetNamespace="http://ipg-online.com/ipgapi/schemas/ipgapi">

   <xs:import namespace="http://ipg-online.com/ipgapi/schemas/v1" schemaLocation="../schemas/v1.xsd" />
   <xs:import namespace="http://ipg-online.com/ipgapi/schemas/a1" schemaLocation="../schemas/a1.xsd" />

   <!-- - - -
```

```
   <!-- Response  - - -
```

```
    </xs:all>
   </xs:complexType>
  </xs:element>

  <xs:element name="EMVCardPresentResponse">
   <xs:complexType>
    <xs:all>
     <xs:element ref="ipgapi:EMVResponseData" />
     <xs:element ref="v1:TLVData" />
    </xs:all>
   </xs:complexType>
  </xs:element>

  <xs:element name="EMVResponseData">
   <xs:complexType>
    <xs:all>
     <xs:element name="IssuerAuthenticationData-91" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:hexBinary">
        <xs:minLength value="8" />
        <xs:maxLength value="16" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
     <xs:element name="IssuerScriptTemplate1-71" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:hexBinary">
        <xs:minLength value="9" />
        <xs:maxLength value="126" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
     <xs:element name="IssuerScriptTemplate2-72" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:hexBinary">
        <xs:minLength value="9" />
        <xs:maxLength value="126" />
       </xs:restriction>
      </xs:simpleType>
```

```
      </xs:element>
      <xs:element name="IssuerAuthorizationResponseCode-8A" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:pattern value="[0-9]{2}" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="MessageControlField-DF4F" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:pattern value="[0-9]{2}" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
     </xs:all>
    </xs:complexType>
   </xs:element>

   <xs:element name="IPGApiActionResponse">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="successfully" type="xs:boolean" />
      <xs:element name="OrderId" type="xs:string" minOccurs="0" />
      <xs:element ref="a1:Error" minOccurs="0" maxOccurs="unbounded" />
      <xs:element name="ResultInfo" type="a1:ResultInfoType" minOccurs="0" />
      <xs:element ref="v1:Basket" minOccurs="0" />
      <xs:element ref="v1:Billing" minOccurs="0" />
      <xs:element ref="a1:Product" minOccurs="0" maxOccurs="unbounded" />
      <xs:element ref="a1:ProductStock" minOccurs="0" maxOccurs="unbounded" />
      <xs:element ref="v1:MandateReference" minOccurs="0" />
      <xs:element ref="v1:Shipping" minOccurs="0" />
      <xs:element ref="a1:TransactionValues" minOccurs="0" maxOccurs="unbounded"
/>
      <xs:element name="RecurringPaymentInformation"
type="a1:RecurringPaymentValues" minOccurs="0" />
      <xs:element name="DataStorageItem" type="a1:DataStorageItem" minOccurs="0"
maxOccurs="unbounded" />
      <xs:element name="ProcessorResponseCode" type="xs:string" minOccurs="0" />
      <xs:element name="ProcessorRequestMessage" type="xs:string" minOccurs="0" />
```

```
      <xs:element name="ProcessorResponseMessage" type="xs:string" minOccurs="0"
/>
      <xs:element name="OrderValues" type="a1:OrderValueType" minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="CardRateForDCC" type="a1:InquiryRateType" minOccurs="0"
maxOccurs="1"/>
      <xs:element name="MerchantRateForDynamicPricing" type="a1:InquiryRateType"
minOccurs="0" maxOccurs="1"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>
</xs:schema>
```

v1.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:v1="http://ipg-
online.com/ipgapi/schemas/v1"
xmlns:cab="http://api.clickandbuy.com/webservices/pay_1_0_0/"
targetNamespace="http://ipg-online.com/ipgapi/schemas/v1"
 elementFormDefault="qualified">

 <xs:import namespace="http://api.clickandbuy.com/webservices/pay_1_0_0/"
schemaLocation="../schemas/ClickAndBuy/PayTypes_1_0_0.xsd" />

 <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - -->
 <!-- main - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - -->
 <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - -->

 <xs:element name="Transaction" type="v1:Transaction" />

 <xs:complexType name="Transaction">
  <xs:sequence>
   <xs:choice>
    <xs:sequence>
     <xs:element name="CreditCardTxType" type="v1:CreditCardTxType" />
     <xs:choice>
      <xs:sequence>
       <xs:element name="CreditCardData" type="v1:CreditCardData" minOccurs="0"
/>
       <xs:element ref="v1:CreditCard3DSecure" minOccurs="0" />
```

```
        </xs:sequence>
        <xs:element ref="v1:EMVCardPresentRequest" />
       </xs:choice>
      </xs:sequence>
      <xs:sequence>
       <xs:element name="CustomerCardTxType" type="v1:CustomerCardTxType" />
       <xs:element name="CustomerCardData" type="v1:CustomerCardData" minOccurs="0"
/>
      </xs:sequence>
      <xs:sequence>
       <xs:element name="DE_DirectDebitTxType" type="v1:DE_DirectDebitTxType" />
       <xs:element name="DE_DirectDebitData" type="v1:DE_DirectDebitData"
minOccurs="0" />
      </xs:sequence>
      <xs:sequence>
       <!-- deprecated -->
       <xs:element name="UK_DebitCardTxType" type="v1:UK_DebitCardTxType" />
       <xs:element name="UK_DebitCardData" type="v1:UK_DebitCardData" minOccurs="0"
/>
      </xs:sequence>
      <xs:sequence>
       <xs:element name="ClickandBuyTxType" type="v1:ClickandBuyTxType" />
       <xs:element name="ClickandBuyData" type="v1:ClickandBuyData" minOccurs="0"
/>
      </xs:sequence>
      <xs:sequence>
       <xs:element name="PayPalTxType" type="v1:PayPalTxType" />
      </xs:sequence>
      <xs:sequence>
       <xs:element name="SofortTxType" type="v1:SofortTxType" />
      </xs:sequence>
      <xs:sequence>
       <xs:element name="TopUpTxType" type="v1:TopUpTxType" />
      </xs:sequence>
     </xs:choice>
     <xs:element ref="v1:Payment" minOccurs="0" />
     <xs:element name="TransactionDetails" type="v1:TransactionDetails"
minOccurs="0" />
     <xs:element ref="v1:Billing" minOccurs="0" />
     <xs:element ref="v1:Shipping" minOccurs="0" />
     <xs:element ref="v1:ClientLocale" minOccurs="0" />
```

```
    <xs:element ref="v1:Basket" minOccurs="0" />
  </xs:sequence>
 </xs:complexType>

 <!-- - - -
```

```
      <xs:element name="Type">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:enumeration value="credit" />
         <xs:enumeration value="forceTicket" />
         <xs:enumeration value="postAuth" />
         <xs:enumeration value="preAuth" />
         <xs:enumeration value="return" />
         <xs:enumeration value="sale" />
         <xs:enumeration value="void" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
     </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="CustomerCardTxType">
  <xs:complexContent>
   <xs:extension base="v1:Options">
    <xs:sequence>
     <xs:element name="Type">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:enumeration value="forceTicket" />
        <xs:enumeration value="BWListCheck" />
        <xs:enumeration value="sale" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="DE_DirectDebitTxType">
  <xs:complexContent>
   <xs:extension base="v1:Options">
```

```
    <xs:sequence>
     <xs:element name="Type">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:enumeration value="credit" />
        <xs:enumeration value="return" />
        <xs:enumeration value="sale" />
        <xs:enumeration value="void" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<!-- deprecated -->
<xs:complexType name="UK_DebitCardTxType">
 <xs:complexContent>
  <xs:extension base="v1:Options">
   <xs:sequence>
    <xs:element name="Type">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:enumeration value="credit" />
       <xs:enumeration value="return" />
       <xs:enumeration value="sale" />
       <xs:enumeration value="void" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
   </xs:sequence>
  </xs:extension>
 </xs:complexContent>
</xs:complexType>

<xs:complexType name="ClickandBuyTxType">
 <xs:complexContent>
  <xs:extension base="v1:Options">
```

```
    <xs:sequence>
     <xs:element name="Type">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:enumeration value="credit" />
        <xs:enumeration value="postAuth" />
        <xs:enumeration value="return" />
        <xs:enumeration value="void" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="PayPalTxType">
  <xs:complexContent>
   <xs:extension base="v1:Options">
    <xs:sequence>
     <xs:element name="Type">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:enumeration value="credit" />
        <xs:enumeration value="postAuth" />
        <xs:enumeration value="return" />
        <xs:enumeration value="void" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
    </xs:sequence>
   </xs:extension>
  </xs:complexContent>
</xs:complexType>

<xs:complexType name="SofortTxType">
  <xs:complexContent>
   <xs:extension base="v1:Options">
    <xs:sequence>
```

```
      <xs:element name="Type">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:enumeration value="return" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
     </xs:sequence>
    </xs:extension>
  </xs:complexContent>
 </xs:complexType>


 <xs:complexType name="Options">
  <xs:sequence>
   <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
  </xs:sequence>
 </xs:complexType>

 <!-- - - -
```

```
        <xs:enumeration value="JCB" />
        <xs:enumeration value="LASER" />
        <xs:enumeration value="MAESTRO" />
        <xs:enumeration value="MASTERCARD" />
        <xs:enumeration value="VISA" />
        <xs:enumeration value="UNIONPAY" />
       </xs:restriction>
     </xs:simpleType>
    </xs:element>
  </xs:sequence>
 </xs:complexType>

 <xs:complexType name="CustomerCardData">
  <xs:sequence>
   <xs:choice>
    <xs:group ref="v1:Card" />
    <xs:element name="TrackData" type="v1:TrackData" />
   </xs:choice>
  </xs:sequence>
 </xs:complexType>

 <xs:complexType name="DE_DirectDebitData">
  <xs:sequence>
   <xs:choice minOccurs="0">
    <xs:sequence>
     <xs:element name="BIC" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:pattern value="[A-Z]{4}DE([A-Z0-9][A-NP-Z0-9])(XXX|[A-WY-Z0-9][A-Z0-
9]{2})?" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
     <xs:element name="IBAN">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:pattern value="DE[0-9]{2}[0-9]{8}[0-9]{10}" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
```

```
      </xs:sequence>
      <xs:element name="TrackData" type="v1:TrackData" />

      <!-- deprecated -->
      <xs:sequence>
       <xs:element name="BankCode">
        <xs:simpleType>
         <xs:restriction base="xs:string">
          <xs:pattern value="([0-9]{8})|([A-Z]{4}DE([A-Z0-9][A-NP-Z0-9])(XXX|[A-
WY-Z0-9][A-Z0-9]{2})?)" />
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
       <xs:element name="AccountNumber">
        <xs:simpleType>
         <xs:restriction base="xs:string">
          <xs:pattern value="([0-9]{1,10})|(DE[0-9]{2}[0-9]{8}[0-9]{10})" />
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
      </xs:sequence>

    </xs:choice>
    <xs:element ref="v1:MandateReference" minOccurs="0" />
    <xs:element ref="v1:MandateType" minOccurs="0" />
   </xs:sequence>
  </xs:complexType>

  <xs:element name="MandateReference">
   <xs:simpleType>
    <xs:restriction base="xs:string">
     <xs:pattern value="[A-Za-z0-9+?/\\:().,'-]{1,35}" />
    </xs:restriction>
   </xs:simpleType>
  </xs:element>

  <xs:element name="MandateType" default="SINGLE">
   <xs:simpleType>
    <xs:restriction base="xs:string">
     <xs:enumeration value="SINGLE" />
```

```
    <xs:enumeration value="RECURRING_COLLECTION" />
   </xs:restriction>
  </xs:simpleType>
</xs:element>

<!-- deprecated -->
<xs:complexType name="UK_DebitCardData">
 <xs:sequence>
  <xs:group ref="v1:Card" />
  <xs:choice>
   <xs:element name="CardCodeValue" type="v1:CardCodeValue" />
   <xs:element name="IssueNo">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]{0,4}" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
  </xs:choice>
 </xs:sequence>
</xs:complexType>

<xs:complexType name="ClickandBuyData">
 <xs:sequence>
  <xs:element minOccurs="0" name="orderDetails" type="cab:OrderDetails">
   <xs:annotation>
    <xs:documentation>details of the payment</xs:documentation>
   </xs:annotation>
  </xs:element>
 </xs:sequence>
</xs:complexType>

<xs:group name="Card">
 <xs:sequence>
  <xs:element name="CardNumber" minOccurs="0">
   <xs:simpleType>
    <xs:restriction base="xs:string">
     <xs:pattern value="[0-9]{13,19}" />
    </xs:restriction>
```

```
      </xs:simpleType>
    </xs:element>
    <xs:element name="ExpMonth">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:pattern value="(0[1-9])|(1[0-2])" />
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
    <xs:element name="ExpYear">
      <xs:simpleType>
        <xs:restriction base="xs:string">
          <xs:pattern value="[0-9]{2}" />
        </xs:restriction>
      </xs:simpleType>
    </xs:element>
  </xs:sequence>
 </xs:group>

 <xs:simpleType name="CardCodeValue">
  <xs:restriction base="xs:string">
   <xs:pattern value="[0-9]{3,4}|" />
  </xs:restriction>
 </xs:simpleType>

 <xs:simpleType name="TrackData">
  <xs:restriction base="xs:string">
   <!-- track 1 length + track 2 length + track 3 length = 90 + 64 + 107 = 261 --
>
   <xs:pattern value="[a-zA-Z0-9;.%=^?/ \\-]{1,261}" />
  </xs:restriction>
 </xs:simpleType>

 <!- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
 <!-- TopUpTxType - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
 <!- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->

 <!-- MobileNetworkServiceProvider -->
```

```
 <xs:element name="MNSP">
  <xs:simpleType>
   <xs:restriction base="xs:string">
    <!-- Deutsche Telekom -->
    <xs:enumeration value="D1" />
    <!-- Vodafon -->
    <xs:enumeration value="D2" />
    <!-- E+ -->
    <xs:enumeration value="EP" />
    <!-- O2 -->
    <xs:enumeration value="VI" />
   </xs:restriction>
  </xs:simpleType>
 </xs:element>

 <xs:element name="PaymentType">
  <xs:simpleType>
   <xs:restriction base="xs:string">
    <xs:enumeration value="Amex" />
    <xs:enumeration value="Cash" />
    <xs:enumeration value="Diners" />
    <xs:enumeration value="ECMC" />
    <xs:enumeration value="JCB" />
    <xs:enumeration value="VISA" />
   </xs:restriction>
  </xs:simpleType>
 </xs:element>

 <!-- - - -
```

```
      <xs:element ref="v1:EMVRequestData" />
      <xs:element ref="v1:TLVData" />
     </xs:choice>
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="CryptData">
  <xs:complexType>
   <xs:all>
    <xs:element name="SRED">
     <xs:complexType>
      <xs:all>
       <xs:element name="Value">
        <xs:simpleType>
         <xs:restriction base="xs:hexBinary">
          <!--
          <xs:length value="256" />
          -->
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
       <xs:element name="KSN">
        <xs:simpleType>
         <xs:restriction base="xs:hexBinary">
          <xs:length value="10" />
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
      </xs:all>
     </xs:complexType>
    </xs:element>
    <xs:element name="PINBlock" minOccurs="0">
     <xs:complexType>
      <xs:all>
       <!--
       <xs:element name="Format">
        <xs:simpleType>
         <xs:restriction base="xs:string">
```

```
            <xs:length value="8" />
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         -->
         <xs:element name="Value">
          <xs:simpleType>
           <xs:restriction base="xs:hexBinary">
            <!--
            <xs:length value="256" />
            -->
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
         <xs:element name="KSN">
          <xs:simpleType>
           <xs:restriction base="xs:hexBinary">
            <xs:length value="10" />
           </xs:restriction>
          </xs:simpleType>
         </xs:element>
        </xs:all>
       </xs:complexType>
     </xs:element>
    </xs:all>
  </xs:complexType>
</xs:element>

<xs:element name="EMVRequestData">
 <xs:complexType>
  <xs:all>
   <xs:element name="AmountAuthorised-9F02" minOccurs="0">
    <xs:simpleType>
     <xs:restriction base="xs:long">
      <xs:pattern value="[0-9]{1,12}" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="AmountOther-9F03" minOccurs="0">
```

```
 <xs:simpleType>
  <xs:restriction base="xs:long">
   <xs:pattern value="[0-9]{1,12}" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="ApplicationCryptogram-9F26" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:length value="8" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="ApplicationIdentifier-4F" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:minLength value="5" />
   <xs:maxLength value="16" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="ApplicationIdentifierTerminal-9F06" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:minLength value="5" />
   <xs:maxLength value="16" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="ApplicationInterchangeProfile-82" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:length value="2" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="ApplicationPANSequenceNumber-5F34" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:int">
```

```
      <xs:minInclusive value="0" />
      <xs:maxInclusive value="99" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="ApplicationTransactionCounter-9F36" minOccurs="0">
    <xs:simpleType>
     <xs:restriction base="xs:hexBinary">
      <xs:length value="2" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="ApplicationVersionNumber-9F09" minOccurs="0">
    <xs:simpleType>
     <xs:restriction base="xs:hexBinary">
      <xs:length value="2" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="CryptogramInformationData-9F27" minOccurs="0">
    <xs:simpleType>
     <xs:restriction base="xs:hexBinary">
      <xs:length value="1" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="CVMResults-9F34" minOccurs="0">
    <xs:simpleType>
     <xs:restriction base="xs:hexBinary">
      <xs:length value="3" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
   <xs:element name="DedicatedFileName-84" minOccurs="0">
    <xs:simpleType>
     <xs:restriction base="xs:hexBinary">
      <xs:minLength value="5" />
      <xs:maxLength value="16" />
     </xs:restriction>
```

```
 </xs:simpleType>
</xs:element>
<xs:element name="InterfaceDeviceSerialNumber-9F1E" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:string">
   <xs:length value="8" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="IssuerApplicationData-9F10" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:maxLength value="32" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="IssuerAuthenticationData-91" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:minLength value="8" />
   <xs:maxLength value="16" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="PointOfServiceEntryMode-9F39" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:int">
   <xs:minInclusive value="0" />
   <xs:maxInclusive value="99" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="TerminalApplicationVersionNumber-9F09" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:length value="2" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
```

```
<xs:element name="TerminalCapabilities-9F33" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:length value="3" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="TerminalCountry-9F1A" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:int">
   <xs:minInclusive value="0" />
   <xs:maxInclusive value="999" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="TerminalType-9F35" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:int">
   <xs:minInclusive value="0" />
   <xs:maxInclusive value="99" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="TerminalVerificationResults-95" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:length value="5" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="TransactionCertificateHashValue-98" minOccurs="0">
 <xs:simpleType>
  <xs:restriction base="xs:hexBinary">
   <xs:length value="20" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>
<xs:element name="TransactionDate-9A" minOccurs="0">
 <xs:simpleType>
```

```
        <xs:restriction base="xs:int">
         <xs:pattern value="[1-9][0-9][0-1][0-9][0-3][0-9]" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="TransactionSequenceCounter-9F41" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:int">
         <xs:pattern value="[0-9]{4,8}" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="TransactionStatusInformation-9B" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:hexBinary">
         <xs:length value="2" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="TransactionType-9C" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:int">
         <xs:minInclusive value="0" />
         <xs:maxInclusive value="99" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="UnpredictableNumber-9F37" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:hexBinary">
         <xs:length value="4" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
     </xs:all>
    </xs:complexType>
   </xs:element>

   <xs:element name="TLVData">
```

```
  <xs:simpleType>
   <xs:restriction base="xs:hexBinary">
    <xs:maxLength value="1024" />
   </xs:restriction>
  </xs:simpleType>
 </xs:element>


 <!-- - - -
```

```
      <xs:element name="AuthenticationValue" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:minLength value="20" />
         <xs:maxLength value="32" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="XID" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:minLength value="20" />
         <xs:maxLength value="32" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
     </xs:all>
    </xs:complexType>
   </xs:element>

   <xs:element name="Payment">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="HostedDataID" type="v1:String128max" minOccurs="0" />
      <xs:element name="HostedDataStoreID" type="v1:String20max" minOccurs="0" />
      <xs:element name="DeclineHostedDataDuplicates" type="xs:boolean"
minOccurs="0" default="false" />
      <xs:group ref="v1:Amount" minOccurs="0" />
     </xs:sequence>
    </xs:complexType>
   </xs:element>

   <xs:group name="Amount">
    <xs:sequence>
     <xs:sequence minOccurs="0">
      <xs:element name="SubTotal" type="v1:AmountValueType" />
      <xs:element name="ValueAddedTax" type="v1:AmountValueType" minOccurs="0" />
      <xs:element name="DeliveryAmount" type="v1:AmountValueType" minOccurs="0" />
     </xs:sequence>
     <!-- ChargeTotal = SubTotal + ValueAddedTax + DeliveryAmount -->
```

```
    <xs:element name="ChargeTotal" type="v1:AmountValueType" />
    <xs:element name="Currency" type="v1:CurrencyType">

    </xs:element>
   </xs:sequence>
  </xs:group>


  <xs:complexType name="TransactionDetails">
   <xs:all>
    <xs:element name="Comments" type="v1:String1024max" minOccurs="0" />
    <xs:element name="InvoiceNumber" type="v1:String48max" minOccurs="0" />
    <xs:element name="DynamicMerchantName" type="v1:String100max" minOccurs="0" />
    <xs:element name="PONumber" type="v1:String128max" minOccurs="0" />
    <xs:element name="OrderId" type="v1:String100max" minOccurs="0" />
    <xs:element ref="v1:OfflineApprovalType" minOccurs="0" />
    <xs:element name="Ip" minOccurs="0">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:pattern value="((25[0-5]|(2[0-4]|1[0-9]|[1-9])?[0-9])\.){3}(25[0-
5]|(2[0-4]|1[0-9]|[1-9])?[0-9])|" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
    <xs:element name="ReferenceNumber" minOccurs="0">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:pattern value="(NEW)?[0-9a-zA-Z]{1,8}" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
    <xs:element name="TDate" minOccurs="0">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:pattern value="[0-9]{10}|" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
    <xs:element ref="v1:TransactionOrigin" minOccurs="0" />
    <xs:element ref="v1:Terminal" minOccurs="0" />
```

```
    <xs:element name="InquiryRateReference" type="v1:InquiryRateReference"
minOccurs="0" />
   </xs:all>
  </xs:complexType>

  <xs:element name="Basket">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="ProductStock" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:enumeration value="check" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
     <xs:element name="Item" maxOccurs="100">
      <xs:complexType>
       <xs:sequence>
        <xs:element name="ID" type="v1:String128max" />
        <xs:element name="Description" type="v1:String128max" />
        <xs:group ref="v1:Amount" />
        <xs:element name="Quantity" type="xs:int" />
        <xs:element name="Option" minOccurs="0" maxOccurs="100">
         <xs:complexType>
          <xs:all>
           <xs:element name="Name" type="v1:String128max" />
           <xs:element name="Choice" type="v1:String128max" />
          </xs:all>
         </xs:complexType>
        </xs:element>
       </xs:sequence>
      </xs:complexType>
     </xs:element>
    </xs:sequence>
   </xs:complexType>
  </xs:element>

  <xs:element name="Terminal">
   <xs:complexType>
    <xs:all>
```

```
    <xs:element name="TerminalID">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:pattern value="[0-9]{8}" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
   </xs:all>
  </xs:complexType>
</xs:element>

<xs:element name="OfflineApprovalType">
 <xs:simpleType>
  <xs:restriction base="xs:string">
   <xs:enumeration value="OFFLINE_APPROVAL" />
   <!-- Fallback: Unable to go online, approval -->
   <xs:enumeration value="FALLBACK" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>

<xs:element name="TransactionOrigin" default="ECI">
 <xs:simpleType>
  <xs:restriction base="xs:string">
   <xs:enumeration value="ECI" />
   <xs:enumeration value="MOTO" />
   <xs:enumeration value="RETAIL" />
  </xs:restriction>
 </xs:simpleType>
</xs:element>

<xs:element name="Billing">
 <xs:complexType>
  <xs:all>
   <xs:element name="CustomerID" type="v1:String32max" minOccurs="0" />
   <xs:element name="Name" type="v1:String96max" minOccurs="0" />
   <xs:element name="Company" type="v1:String96max" minOccurs="0" />
   <xs:element name="Address1" type="v1:String96max" minOccurs="0" />
   <xs:element name="Address2" type="v1:String96max" minOccurs="0" />
```

```
      <xs:element name="City" type="v1:String96max" minOccurs="0" />
      <xs:element name="State" type="v1:String96max" minOccurs="0" />
      <xs:element name="Zip" type="v1:String24max" minOccurs="0" />
      <xs:element name="Country" type="v1:String32max" minOccurs="0" />
      <xs:element name="Phone" type="v1:String32max" minOccurs="0" />
      <xs:element name="Fax" type="v1:String32max" minOccurs="0" />
      <xs:element name="Email" type="v1:String64max" minOccurs="0" />

      <!-- DEPRECATED, this field is not evaluated and will be removed in v2.xsd --
>
      <xs:element name="Addrnum" type="v1:String96max" minOccurs="0" />
    </xs:all>
  </xs:complexType>
 </xs:element>

 <xs:element name="Shipping">
  <xs:complexType>
   <xs:all>
    <!-- DEPRECATED, this field is not evaluated and will be removed in v2.xsd --
>
     <xs:element name="Type" type="v1:String20max" minOccurs="0" />

     <xs:element name="Name" type="v1:String96max" minOccurs="0" />
     <xs:element name="Address1" type="v1:String96max" minOccurs="0" />
     <xs:element name="Address2" type="v1:String96max" minOccurs="0" />
     <xs:element name="City" type="v1:String96max" minOccurs="0" />
     <xs:element name="State" type="v1:String96max" minOccurs="0" />
     <xs:element name="Zip" type="v1:String24max" minOccurs="0" />
     <xs:element name="Country" type="v1:String32max" minOccurs="0" />
    </xs:all>
  </xs:complexType>
 </xs:element>

 <xs:element name="ClientLocale">
  <xs:complexType>
   <xs:all>
    <xs:element name="Language">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <!-- ISO 639-1 -->
```

```
        <xs:pattern value="[a-z]{2}" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
     <xs:element name="Country" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <!-- ISO 3166 -->
        <xs:pattern value="[A-Z]{2}|" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
    </xs:all>
   </xs:complexType>
  </xs:element>


  <!-- - - -
```

```
  <xs:restriction base="xs:string">
    <xs:maxLength value="4000" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String1024max">
  <xs:restriction base="xs:string">
    <xs:maxLength value="1024" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String128max">
  <xs:restriction base="xs:string">
    <xs:maxLength value="128" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String100max">
  <xs:restriction base="xs:string">
    <xs:maxLength value="100" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String96max">
  <xs:restriction base="xs:string">
    <xs:maxLength value="96" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String64max">
  <xs:restriction base="xs:string">
    <xs:maxLength value="64" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String50max">
  <xs:restriction base="xs:string">
    <xs:maxLength value="50" />
  </xs:restriction>
```

```
</xs:simpleType>

<xs:simpleType name="String48max">
 <xs:restriction base="xs:string">
  <xs:maxLength value="48" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String32max">
 <xs:restriction base="xs:string">
  <xs:maxLength value="32" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String30max">
 <xs:restriction base="xs:string">
  <xs:maxLength value="30" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String24max">
 <xs:restriction base="xs:string">
  <xs:maxLength value="24" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String20max">
 <xs:restriction base="xs:string">
  <xs:maxLength value="20" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String10max">
 <xs:restriction base="xs:string">
  <xs:maxLength value="10" />
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="String6max">
```

```
    <xs:restriction base="xs:string">
      <xs:maxLength value="6" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="String3max">
    <xs:restriction base="xs:string">
      <xs:maxLength value="3" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="CurrencyType">
    <xs:restriction base="xs:string">
      <!-- http://de.wikipedia.org/wiki/ISO_4217 -->
      <!-- http://en.wikipedia.org/wiki/ISO_4217 -->
      <xs:pattern value="[0-9]{3}|[A-Z]{3}" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="AmountValueType">
    <xs:restriction base="xs:decimal">
      <xs:fractionDigits value="3" />
      <xs:totalDigits value="12" />
      <xs:minInclusive value="0" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="BINType">
    <xs:restriction base="xs:int">
      <xs:totalDigits value="6" />
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

a1.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:a1="http://ipg-online.com/ipgapi/schemas/a1" xmlns:v1="http://ipg-online.com/ipgapi/schemas/v1" xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi"
 targetNamespace="http://ipg-online.com/ipgapi/schemas/a1" elementFormDefault="qualified">

 <xs:import namespace="http://ipg-online.com/ipgapi/schemas/ipgapi" schemaLocation="../schemas/ipgapi.xsd" />
 <xs:import namespace="http://ipg-online.com/ipgapi/schemas/v1" schemaLocation="../schemas/v1.xsd" />

 <!-- - - -
```

```
    </xs:sequence>
   </xs:complexType>
  </xs:element>

  <!-- - - -
```

```
      <xs:element name="Email" type="v1:String64max" minOccurs="0" />
     </xs:all>
    </xs:complexType>
   </xs:element>

   <xs:element name="Validate">
    <xs:complexType>
     <xs:sequence>
      <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
      <xs:choice>
       <xs:element name="CreditCardData" type="v1:CreditCardData" minOccurs="0" />
       <xs:element name="CustomerCardData" type="v1:CustomerCardData" minOccurs="0"
/>
       <xs:element name="DE_DirectDebitData" type="v1:DE_DirectDebitData"
minOccurs="0" />
      </xs:choice>
      <xs:element ref="v1:Payment" minOccurs="0" />
      <xs:element name="TransactionDetails" type="v1:TransactionDetails"
minOccurs="0" />
      <xs:element ref="v1:Billing" minOccurs="0" />
     </xs:sequence>
    </xs:complexType>
   </xs:element>

   <xs:element name="GetExternalTransactionStatus">
    <xs:complexType>
     <xs:all>
      <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
      <xs:element name="OrderId" type="v1:String100max" />
      <xs:element name="TDate" minOccurs="0">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:pattern value="[0-9]{10}|" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
     </xs:all>
    </xs:complexType>
   </xs:element>
```

```
 <xs:element name="GetExternalConsumerInformation">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
    <xs:element name="OrderId" type="v1:String100max" minOccurs="0" />
    <xs:element name="DataProvider">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:enumeration value="Bürgel" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
    <xs:element name="FirstName" type="v1:String30max" />
    <xs:element name="Surname" type="v1:String30max" />
    <xs:element name="Birthday" type="a1:StringDate" />
    <xs:element name="Street" type="v1:String30max" />
    <xs:element name="HouseNumber" type="v1:String10max" />
    <xs:element name="PostCode" type="v1:String10max" />
    <xs:element name="City" type="v1:String30max" />
    <xs:element name="Country">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <!-- ISO 3166 -->
       <xs:pattern value="[0-9]{3}" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
    <xs:element name="DisplayProcessorMessages" type="xs:boolean" minOccurs="0"
default="true" />
   </xs:sequence>
  </xs:complexType>
 </xs:element>

 <xs:element name="GetLastOrders">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
    <xs:element name="Count" default="10" minOccurs="0">
     <xs:simpleType>
      <xs:restriction base="xs:positiveInteger">
```

```
        <xs:maxInclusive value="100" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
     <xs:element name="DateFrom" type="xs:dateTime" minOccurs="0" />
     <xs:element name="DateTo" type="xs:dateTime" minOccurs="0" />
     <xs:element name="OrderID" type="v1:String100max" minOccurs="0" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="GetLastTransactions">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
    <xs:element name="count" default="10" minOccurs="0">
     <xs:simpleType>
      <xs:restriction base="xs:positiveInteger">
       <xs:maxInclusive value="100" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
    <xs:group ref="a1:TransactionID" minOccurs="0" />
   </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:group name="TransactionID">
  <xs:sequence>
   <xs:element name="OrderId" type="v1:String100max" />
   <xs:element name="TDate">
    <xs:simpleType>
     <xs:restriction base="xs:string">
      <xs:pattern value="[0-9]{10}|" />
     </xs:restriction>
    </xs:simpleType>
   </xs:element>
  </xs:sequence>
</xs:group>
```

```
 <xs:element name="RequestCardRateForDCC">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="StoreId" type="v1:String20max" />
    <xs:element name="BIN" type="v1:BINType" />
    <xs:element name="BaseAmount" type="v1:AmountValueType" minOccurs="0" />
   </xs:sequence>
  </xs:complexType>
 </xs:element>

 <xs:element name="RequestMerchantRateForDynamicPricing">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="StoreId" type="v1:String20max" />
    <xs:element name="ForeignCurrency" type="v1:CurrencyType" />
    <xs:element name="BaseAmount" type="v1:AmountValueType" minOccurs="0" />
   </xs:sequence>
  </xs:complexType>
 </xs:element>
```

```
 <!-- - - -
```

```
 <xs:complexType name="DataStorageItem">
  <xs:sequence>
   <xs:choice>
    <xs:element name="CreditCardData" type="v1:CreditCardData" />
    <xs:element name="DE_DirectDebitData" type="v1:DE_DirectDebitData" />
    <xs:element name="OrderId" type="v1:String100max" />
    <xs:element name="Function">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:enumeration value="delete" />
       <xs:enumeration value="display" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
   </xs:choice>
   <xs:element name="HostedDataID" type="v1:String128max" />
   <xs:element name="DeclineHostedDataDuplicates" type="xs:boolean" minOccurs="0"
default="false" />
  </xs:sequence>
 </xs:complexType>


 <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
 <!-- Product Inventory - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
 <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->


 <xs:element name="ManageProducts">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
    <xs:element name="Function">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:enumeration value="delete" />
       <xs:enumeration value="display" />
       <xs:enumeration value="store" />
      </xs:restriction>
     </xs:simpleType>
```

```
      </xs:element>
      <xs:element ref="a1:Product" maxOccurs="unbounded" />
     </xs:sequence>
   </xs:complexType>
  </xs:element>

  <xs:element name="Product">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="ProductID" type="v1:String50max" />
     <xs:element name="Description" type="v1:String4000max" minOccurs="0" />
     <xs:element name="OfferStarts" type="xs:dateTime" minOccurs="0" />
     <xs:element name="OfferEnds" type="xs:dateTime" minOccurs="0" />
     <xs:group ref="v1:Amount" minOccurs="0" />
     <xs:element name="Choice" type="a1:ProductChoice" minOccurs="0"
maxOccurs="99"/>
    </xs:sequence>
   </xs:complexType>
  </xs:element>

  <xs:complexType name="ProductChoice">
   <xs:all>
    <xs:element name="OptionName" type="v1:String128max" />
    <xs:element name="Name" type="v1:String128max" />
   </xs:all>
  </xs:complexType>

  <xs:element name="ManageProductStock">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
     <xs:element name="Function">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:enumeration value="add" />
        <xs:enumeration value="delete" />
        <xs:enumeration value="display" />
        <xs:enumeration value="subtract" />
       </xs:restriction>
      </xs:simpleType>
```

```
      </xs:element>
      <xs:element ref="a1:ProductStock" maxOccurs="unbounded" />
     </xs:sequence>
   </xs:complexType>
  </xs:element>

  <xs:element name="ProductStock">
   <xs:complexType>
    <xs:sequence>
     <xs:element name="ProductID" type="v1:String50max" />
     <xs:element name="Choice" type="a1:ProductChoice" minOccurs="0"
maxOccurs="3"/>
     <xs:element name="Quantity" type="xs:nonNegativeInteger" minOccurs="0" />
    </xs:sequence>
   </xs:complexType>
  </xs:element>

  <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
  <!-- RecurringPayment  - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
  <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->

  <xs:complexType name="RecurringPayment">
   <xs:all>
    <xs:element name="Function" type="a1:Function" />
    <xs:element name="OrderId" type="v1:String100max" minOccurs="0" />
    <xs:element name="StoreId" type="v1:String20max" minOccurs="0" />
    <xs:element name="Comments" type="v1:String1024max" minOccurs="0" />
    <xs:element name="InvoiceNumber" type="v1:String48max" minOccurs="0" />
    <xs:element name="DynamicMerchantName" type="v1:String100max" minOccurs="0" />
    <xs:element name="PONumber" type="v1:String128max" minOccurs="0" />
    <xs:element name="RecurringPaymentInformation"
type="a1:RecurringPaymentInformation" minOccurs="0" />
    <xs:element name="CreditCardData" type="v1:CreditCardData" minOccurs="0" />
    <xs:element name="DE_DirectDebitData" type="v1:DE_DirectDebitData"
minOccurs="0" />
    <xs:element ref="v1:MandateReference" minOccurs="0" />
    <xs:element name="ReferencedOrderId" type="v1:String100max" minOccurs="0" />
    <xs:element ref="v1:Payment" minOccurs="0" />
```

```
    <xs:element ref="v1:Basket" minOccurs="0" />
    <xs:element ref="v1:Billing" minOccurs="0" />
    <xs:element ref="v1:CreditCard3DSecure" minOccurs="0" />
    <xs:element ref="v1:Shipping" minOccurs="0" />
    <xs:element name="Ip" minOccurs="0">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:pattern value="(25[0-5]|(2[0-4]|1[0-9]|[1-9])?[0-9])\.(25[0-5]|(2[0-
4]|1[0-9]|[1-9])?[0-9])\.(25[0-5]|(2[0-4]|1[0-9]|[1-9])?[0-9])\.(25[0-5]|(2[0-
4]|1[0-9]|[1-9])?[0-9])|" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
    <xs:element name="TransactionOrigin" default="ECI" minOccurs="0">
     <xs:simpleType>
      <xs:restriction base="xs:string">
       <xs:enumeration value="ECI" />
       <xs:enumeration value="MOTO" />
       <xs:enumeration value="RETAIL" />
      </xs:restriction>
     </xs:simpleType>
    </xs:element>
   </xs:all>
  </xs:complexType>

  <xs:simpleType name="Function">
   <xs:restriction base="xs:string">
    <xs:enumeration value="cancel" />
    <xs:enumeration value="install" />
    <xs:enumeration value="modify" />
    <xs:enumeration value="perform only in test environment" />
   </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="RecurringPaymentInformation">
   <xs:sequence>
    <xs:element name="RecurringStartDate" type="a1:StringDate" minOccurs="0" />
    <xs:element name="InstallmentCount" minOccurs="0">
     <xs:simpleType>
      <xs:restriction base="xs:integer">
```

```
        <xs:maxInclusive value="999" />
        <xs:minInclusive value="1" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>

     <!-- for future use -->
     <xs:element name="MaximumFailures" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:integer">
        <xs:maxInclusive value="5" />
        <xs:minInclusive value="1" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>

     <xs:element name="InstallmentFrequency" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:integer">
        <xs:maxInclusive value="1000" />
        <xs:minInclusive value="1" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
     <xs:element name="InstallmentPeriod" minOccurs="0">
      <xs:simpleType>
       <xs:restriction base="xs:string">
        <xs:enumeration value="day" />
        <xs:enumeration value="week" />
        <xs:enumeration value="month" />
        <xs:enumeration value="year" />
       </xs:restriction>
      </xs:simpleType>
     </xs:element>
    </xs:sequence>
   </xs:complexType>

   <!-- - - -
```

```
 <!-- Response - - -
```

```
    <xs:complexContent>
     <xs:extension base="v1:Transaction">
      <xs:sequence>
       <xs:element ref="ipgapi:IPGApiOrderResponse" minOccurs="0" />
       <xs:element name="ReceiptNumber" type="xs:string" minOccurs="0" />
       <xs:element name="ResponseCode" type="xs:string" minOccurs="0" />
       <xs:element name="TraceNumber" type="xs:string" minOccurs="0" />
       <xs:element name="TransactionState" type="xs:string" minOccurs="0" />
       <xs:element name="UserID" type="xs:string" minOccurs="0" />
       <xs:element name="GiroPayTxType" type="v1:DE_DirectDebitTxType"
minOccurs="0" />
       <xs:element name="GiroPayData" type="v1:DE_DirectDebitData" minOccurs="0"
/>
       <xs:element name="SubmissionComponent" type="xs:string" minOccurs="0" />
      </xs:sequence>
     </xs:extension>
    </xs:complexContent>
   </xs:complexType>
  </xs:element>

  <xs:complexType name="RecurringPaymentValues">
   <xs:complexContent>
    <xs:extension base="a1:RecurringPaymentInformation">
     <xs:sequence>
      <xs:element name="State">
       <xs:simpleType>
        <xs:restriction base="xs:string">
         <xs:enumeration value="Cancelled" />
         <xs:enumeration value="Inactivated" />
         <xs:enumeration value="Installed" />
        </xs:restriction>
       </xs:simpleType>
      </xs:element>
      <xs:element name="CreationDate" type="a1:StringDate" />
      <xs:element name="FailureCount" type="xs:int" minOccurs="0" />
      <xs:element name="NextAttemptDate" type="a1:StringDate" />
      <xs:element name="RunCount" type="xs:int" minOccurs="0" />
      <xs:element name="CreditCardData" type="v1:CreditCardData" minOccurs="0" />
      <xs:element name="DE_DirectDebitData" type="v1:DE_DirectDebitData"
minOccurs="0" />
```

```
      <xs:element name="HostedDataID" type="v1:String128max" minOccurs="0" />
      <xs:element name="HostedDataStoreID" type="v1:String20max" minOccurs="0" />
      <xs:group ref="v1:Amount" minOccurs="0" />
      <xs:element ref="v1:TransactionOrigin" minOccurs="0" />
      <xs:element name="InvoiceNumber" type="v1:String48max" minOccurs="0" />
      <xs:element name="PONumber" type="v1:String128max" minOccurs="0" />
      <xs:element name="Comments" type="v1:String1024max" minOccurs="0" />
     </xs:sequence>
    </xs:extension>
   </xs:complexContent>
  </xs:complexType>


  <xs:complexType name="InquiryRateType">
   <xs:sequence>
    <xs:element ref="v1:InquiryRateId" />
    <xs:element name="ForeignCurrencyCode" type="v1:CurrencyType" minOccurs="0"
maxOccurs="1" />
    <xs:element name="ForeignAmount" type="v1:AmountValueType" minOccurs="0"
maxOccurs="1" />
    <xs:element name="ExchangeRate" type="xs:decimal" minOccurs="0" maxOccurs="1"
/>
    <xs:element name="DccApplied" type="xs:boolean" minOccurs="0" maxOccurs="1" />
    <xs:element name="DccOffered" type="xs:boolean" minOccurs="0" maxOccurs="1" />
    <xs:element name="ExpirationTimestamp" type="xs:dateTime" minOccurs="0"
maxOccurs="1" />
    <xs:element name="MarginRatePercentage" type="xs:decimal" minOccurs="0"
maxOccurs="1" />
    <xs:element name="ExchangeRateSourceName" type="v1:String32max" minOccurs="0"
maxOccurs="1" />
     <xs:element name="ExchangeRateSourceTimestamp" type="xs:dateTime"
minOccurs="0" maxOccurs="1" />
   </xs:sequence>
  </xs:complexType>


  <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
  <!-- simple types - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
  <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
```

```
 <xs:simpleType name="StringDate">
  <xs:restriction base="xs:string">
   <!-- YYYYMMDD -->
   <xs:pattern value="[0-9]{8}" />
  </xs:restriction>
 </xs:simpleType>

</xs:schema>
```

Fehleranalyse - Merchant Exceptions

Premature end of message

```
<detail>
      XML is not wellformed: Premature end of message.
</detail>
```

Mögliche Erklärung:

Sie haben eine leere Nachricht gesendet.

Content is not allowed in prolog

```
<detail>
      XML is not wellformed: Content is not allowed in prolog.
</detail>
```

Mögliche Erklärung:

Die Nachricht kann nicht als XML interpretiert werden.

document structures must start and end within the same entity

```
<detail>
      XML is not wellformed:
      XML document structures must start and end within the same entity.
</detail>
```

Mögliche Erklärung:

Die Nachricht beginnt wie eine XML Nachricht, enthält jedoch kein End-Tag nach dem ersten Open-Tag.

The element type "SOAP-ENV:Body" must be terminated

```
<detail>
      XML is not wellformed:
      The element type "SOAP-ENV:Body" must be terminated
      by the matching end-tag "&lt;/SOAP-ENV:Body&gt;".
</detail>
```

Mögliche Erklärung:

Ein End-Tag zu einem Open-Tag fehlt. Im obigen Beispiel fehlt das End-Tag </SOAP-ENV:Body>.

Element type "irgend" must be followed by either attribute…

```
<detail>
      XML is not wellformed:
      Element type "irgend" must be followed by either attribute specifications,
      "&gt;" or "/&gt;".
</detail>
```

Mögliche Erklärung:

Bei der Nachricht handelt es sich nicht um korrektes XML.

Open quote is expected for attribute "xmlns:ns3"…

```
<detail>
      XML is not wellformed:
      Open quote is expected for attribute "xmlns:ns3"
      associated with an element type "ns3:IPGApiOrderRequest".
</detail>
```

Mögliche Erklärung:

Der Wert eines Attributs ist nicht in Anführungszeichen.

The prefix "ipgapi" for element "ipgapi:…

```
<detail>
      XML is not wellformed:
      The prefix "ipgapi" for element "ipgapi:IPGApiOrderRequest"
      is not bound.
</detail>
```

Mögliche Erklärung:

Der Namensraum “ipgapi” ist nicht deklariert. Um einen Namensraum zu deklarieren, muss das Präfix xmlns verwendet werden, z. B.

`xmlns:ipgapi=”http://ipg-online.com/ipgapi/schemas/ipgapi” as attribute in the top level tag of the IPG API message (IPGApiOrderRequest or IPGApiActionRequest).`

The prefix "xmln" for attribute "xmln:ns2"…

```
<detail>
      XML is not wellformed:
      The prefix "xmln" for attribute "xmln:ns2" associated
      with an element type "ns3:IPGApiOrderRequest" is not bound.
</detail>
```

Mögliche Erklärung:

Um einen eigenen Namensraum zu deklarieren ist nur xmlns erlaubt. In diesem Fall wurde xmln verwendet.

Unable to create envelope from given source…

```
<detail>
      XML is not wellformed:
      Unable to create envelope from given source
      because the namespace was not recognized
</detail>
```

Mögliche Erklärung:

Die XML Nachricht sowie die SOAP Nachricht sind korrekt, jedoch hat die IPG API Nachricht im Body keinen oder falsch deklarierte Namensräume. Die korrekten Namensräume sind in der xsd beschrieben.

The processing instruction target matching "[xX]…

```
<detail>
      XML is not wellformed:
      The processing instruction target matching "[xX][mM][lL]"
      is not allowed.
</detail>
```

Mögliche Erklärung:

Die gesamte Nachricht muss eine korrekte XML Nachricht sein, sodass die IPG API Nachricht kein <?xml … ?> enthalten darf.

Unexpected characters before XML declaration

```
<detail>
      Unexpected characters before XML declaration
</detail>
```

Mögliche Erklärung:

Die XML Nachricht muss mit “<?xml” beginnen. Bitte überprüfen Sie, ob ggf. eine Leerzeile oder ein Leerzeichen zu Beginn stehen.

Unable to create envelope from given source…

```
<detail>
      XML is not a SOAP message:
      Unable to create envelope from given source
      because the root element is not named "Envelope"
</detail>
```

Mögliche Erklärung:

Die SOAP Nachricht ist nicht korrekt.

Error with the determination of the type

```
<detail>
      XML is not a valid SOAP message:
      Error with the determination of the type.
      Probably the envelope part is not correct.
</detail>
```

Mögliche Erklärung:

Das SOAP Body-Tag fehlt.

Source object passed to "{0}" has no contents

```
<detail>
```

```
      Source object passed to ''{0}'' has no contents.
</detail>
```

Mögliche Erklärung:

Das SOAP Body-Tag ist leer.

unsupported top level {namespace}tag…

```
<detail>
      Included XML is not a valid IPG API message:
      unsupported top level {namespace}tag "irgendwas" in the soap body. Only one
      of [
      {http://ipg-online.com/ipgapi/schemas/ipgapi}IPGApiActionRequest,
      {http://ipg-online.com/ipgapi/schemas/ipgapi}IPGApiOrderRequest
      ] allowed.
</detail>
```

Mögliche Erklärung:

Das erste Tag der IPG API Nachricht muss einem IPGApiActionRequest oder IPGApiOrderRequest entsprechen. Im obigen Beispiel hat das Tag keinen Namensraum.

unsupported top level {namespace}tag "{http…

```
<detail>
      Included XML is not a valid IPG API message:
      unsupported top level {namespace}tag
      "{http://firstdata.de/ipgapi/schemas/ipgapi}IPGApiOrderRequest" in the soap
      body. Only one of [
      {http://ipg-online.com/ipgapi/schemas/ipgapi}IPGApiActionRequest,
      {http://ipg-online.com/ipgapi/schemas/ipgapi}IPGApiOrderRequest
      ] allowed.
</detail>
```

Mögliche Erklärung:

Das Top-Level Tag entspricht keinem zugelassenen Tag. In diesem Fall ist der Namensraum falsch.

cvc-pattern-valid:

```
<detail>
      cvc-pattern-valid:
      Value '1.234' is not facet-valid with respect to pattern
      '([1-9]([0-9]{0,12}))?[0-9](\.[0-9]{1,2})?' for type
      '#AnonType_ChargeTotalAmount'
      cvc-type.3.1.3:
      The value '1.234' of element 'ns3:ChargeTotal' is not valid.
</detail>
```

Mögliche Erklärung:

Der Wert eines Tags entspricht nicht der Deklaration in der xsd. Im Beispiel ist der Wert dreistellig obwohl nur zwei Stellen erlaubt sind.

cvc-complex-type.2.4.a:

```
<detail>
```

```
      cvc-complex-type.2.4.a:
      Invalid content was found starting with element 'ns2:ExpYear'.
      One of '{"http://ipg-online.com/ipgapi/schemas/v1":ExpMonth}'
      is expected.
</detail>
```

Mögliche Erklärung:

Die Reihenfolge der Tags entspricht nicht der Deklaration in der xsd. Im Beispiel wird der Monat des Ablaufdatums anstatt des Jahres erwartet.

Fehleranalyse - Processing Exceptions

```
<detail>
      <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
            <ipgapi:CommercialServiceProvider />
            <ipgapi:TransactionTime>1233656751183</ipgapi:TransactionTime>
            <ipgapi:ProcessorReferenceNumber />
            <ipgapi:ProcessorResponseMessage />
            <ipgapi:ErrorMessage>
                  SGS-C: 000003:
                  illegal combination of values for the 3DSecure:
                  (VerificationResponse, PayerAuthenticationResponse,
                  PayerAuthenticationCode) N Y null
            </ipgapi:ErrorMessage>
            <ipgapi:OrderId />
            <ipgapi:ApprovalCode />
            <ipgapi:AVSResponse />
            <ipgapi:TDate />
            <ipgapi:TransactionResult>FAILED</ipgapi:TransactionResult>
            <ipgapi:TerminalID />
            <ipgapi:ProcessorResponseCode />
            <ipgapi:ProcessorApprovalCode />
            <ipgapi:ProcessorReceiptNumber />
            <ipgapi:ProcessorTraceNumber />
      </ipgapi:IPGApiOrderResponse>
</detail>
```

Erklärung:

Die Kombination der drei Werte VerificationResponse, PayerAuthenticationResponse und PayerAuthenticationCode für 3DSecure sind falsch. Zugelassene Kombinationen sind:

Verification-Response	Payer-Authentication-Response	Payer-Authentication-Code	IPG 3dsecure response code	Kommentar
null	null	null	n/a	Transaktion wird ohne 3D Secure Informationen zur Autorisierung weitergeleitet Kein MC ECI, Visa ECI = 7
N	null	null	7	Karteninhaber nicht aktiviert Kein MC ECI, Visa ECI = 6
N	N	null	7	Karteninhaber nicht aktiviert Kein MC ECI, Visa ECI = 6
U	null	null	5	Authentifizierung nicht möglich (DS nicht erreichbar) Kein MC ECI, Visa ECI = 7
Y	A	null	4	Authentifizierungsversuch (Keine Auskunft zum Ergebnis vom ACS) MC ECI = 1, Visa ECI = 6
Y	A	x	4	Authentifizierungsversuch (Keine Auskunft zum Ergebnis vom ACS) MC ECI = 1, Visa ECI = 6

Y	U	null		Authentifizierung nicht möglich (ACS nicht erreichbar)
			6	No MC ECI, Visa ECI = 7
Y	Y	null		Erfolgreiche Authentifizierung (kein CAAV / UCAF)
			2	MC ECI = 2, Visa ECI = 5
Y	Y	x		Erfolgreiche Authentifizierung
			1	MC ECI = 2, Visa ECI = 5
Y	N	null		Authentifizierung negativ – Die Transaktion wird abgelehnt. ("N:-5101:3D Secure authentication failed")
			3	No MC or Visa ECI

Andere Kombinationen werden abgelehnt.

Die vom Merchant Plug-in generierte XID muss für Visa Transaktionen gesetzt werden.

Bitte entnehmen Sie Details zu diesen Feldern den 3D Secure Dokumentationen der Kreditkartenorganisationen.

```
<detail>
	<ipgapi:IPGApiOrderResponse
		xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
		<ipgapi:CommercialServiceProvider />
		<ipgapi:TransactionTime>1233659493267</ipgapi:TransactionTime>
		<ipgapi:ProcessorReferenceNumber />
		<ipgapi:ProcessorResponseMessage />
		<ipgapi:ErrorMessage>
			SGS-005002:
			The merchant is not setup to support the requested service.
		</ipgapi:ErrorMessage>
		<ipgapi:OrderId>
			IPGAPI-REQUEST-9c555d62-3850-4726-8589-5a2444c98c5d
		</ipgapi:OrderId>
		<ipgapi:ApprovalCode />
		<ipgapi:AVSResponse />
		<ipgapi:TDate />
		<ipgapi:TransactionResult>FAILED</ipgapi:TransactionResult>
		<ipgapi:TerminalID />
		<ipgapi:ProcessorResponseCode />
		<ipgapi:ProcessorApprovalCode />
		<ipgapi:ProcessorReceiptNumber />
		<ipgapi:ProcessorTraceNumber />
	</ipgapi:IPGApiOrderResponse>
</detail>
```

Erklärung:

In diesem Beispiel wurde versucht eine Direct Debit Transaktion zu senden, jedoch ist der Händler für diese Zahlungsart nicht registriert. Sollten Sie dieses Ergebnis für eine Zahlart bekommen, die Sie beauftragt haben, kontaktieren Sie bitte unser technsiches Support-Team.

```
<detail>
	<ipgapi:IPGApiOrderResponse
		xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
		<ipgapi:CommercialServiceProvider />
		<ipgapi:TransactionTime>1233656752933</ipgapi:TransactionTime>
		<ipgapi:ProcessorReferenceNumber />
		<ipgapi:ProcessorResponseMessage />
		<ipgapi:ErrorMessage>
			SGS-005005: Duplicate transaction.
		</ipgapi:ErrorMessage>
		<ipgapi:OrderId>
			IPGAPI-REQUEST-29351d8e-2634-4725-9d93-91b83704e00d
		</ipgapi:OrderId>
		<ipgapi:ApprovalCode />
		<ipgapi:AVSResponse />
		<ipgapi:TDate />
		<ipgapi:TransactionResult>FRAUD</ipgapi:TransactionResult>
		<ipgapi:TerminalID />
		<ipgapi:ProcessorResponseCode />
		<ipgapi:ProcessorApprovalCode />
		<ipgapi:ProcessorReceiptNumber />
		<ipgapi:ProcessorTraceNumber />
	</ipgapi:IPGApiOrderResponse>
</detail>
```

Erklärung:

Es wurde merhrmals hintereinander versucht, die gleiche Transaktion zu senden. Bitte beachten Sie zur Einrichtung der Sperrdauer die Hinweise im Benutzerhandbuch Virtual Terminal.

```
<detail>
	<ipgapi:IPGApiOrderResponse
		xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
		<ipgapi:CommercialServiceProvider />
		<ipgapi:TransactionTime>1233656752308</ipgapi:TransactionTime>
		<ipgapi:ProcessorReferenceNumber />
		<ipgapi:ProcessorResponseMessage />
		<ipgapi:ErrorMessage>
			SGS-005009:
			The currency is not allowed for this terminal.
		</ipgapi:ErrorMessage>
		<ipgapi:OrderId>
			IPGAPI-REQUEST-a58f6631-eb71-49c8-bbca-23fff53252fc
		</ipgapi:OrderId>
		<ipgapi:ApprovalCode />
		<ipgapi:AVSResponse />
		<ipgapi:TDate />
		<ipgapi:TransactionResult>FAILED</ipgapi:TransactionResult>
		<ipgapi:TerminalID />
		<ipgapi:ProcessorResponseCode />
		<ipgapi:ProcessorApprovalCode />
		<ipgapi:ProcessorReceiptNumber />
		<ipgapi:ProcessorTraceNumber />
	</ipgapi:IPGApiOrderResponse>
</detail>
```

Erklärung:

In diesem Beispiel versucht der Händler eine Transaktion in US-Dollar einzureichen, ist jedoch für diese Währung nicht registriert. Sollten Sie dieses Ergebnis für eine Währung

bekommen, die Sie beauftragt haben, kontaktieren Sie bitte unser technsiches Support-Team.

```
<detail>
      <ipgapi:IPGApiOrderResponse xmlns:ipgapi="http://ipg-
online.com/ipgapi/schemas/ipgapi">
            <ipgapi:CommercialServiceProvider />
            <ipgapi:TransactionTime>1235390834046</ipgapi:TransactionTime>
            <ipgapi:ProcessorReferenceNumber />
            <ipgapi:ProcessorResponseMessage />
            <ipgapi:ErrorMessage>
                  SGS-005999: There was an unknown error in the database.
            </ipgapi:ErrorMessage>
            <ipgapi:OrderId>
                  IPGAPI-REQUEST-d26ea5c1-d0de-41d2-8c41-3d6755fc204c
            </ipgapi:OrderId>
            <ipgapi:ApprovalCode />
            <ipgapi:AVSResponse />
            <ipgapi:TDate>1235390830</ipgapi:TDate>
            <ipgapi:TransactionResult>DECLINED</ipgapi:TransactionResult>
            <ipgapi:TerminalID />
            <ipgapi:ProcessorResponseCode />
            <ipgapi:ProcessorApprovalCode />
            <ipgapi:ProcessorReceiptNumber />
            <ipgapi:ProcessorTraceNumber />
      </ipgapi:IPGApiOrderResponse>
</detail>
```

Erklärung:

Diese Fehlermeldung tritt beispielsweise auf, wenn beim Storno einer Kreditkarten-Transaktion eine abweichende Zahlart (Direct Debit) angegeben wurde.

```
<detail>
      <ipgapi:IPGApiOrderResponse
            xmlns:ipgapi="http://ipg-online.com/ipgapi/schemas/ipgapi">
            <ipgapi:CommercialServiceProvider />
            <ipgapi:TransactionTime>1234346305732</ipgapi:TransactionTime>
            <ipgapi:ProcessorReferenceNumber />
            <ipgapi:ProcessorResponseMessage />
            <ipgapi:ErrorMessage>
                  SGS-032000: Unknown processor error occured.
            </ipgapi:ErrorMessage>
            <ipgapi:OrderId>
                  IPGAPI-REQUEST-b3223ee5-156b-4d22-bc3f-910709d59202
            </ipgapi:OrderId>
            <ipgapi:ApprovalCode />
            <ipgapi:AVSResponse />
            <ipgapi:TDate>1234346284</ipgapi:TDate>
            <ipgapi:TransactionResult>DECLINED</ipgapi:TransactionResult>
            <ipgapi:TerminalID />
            <ipgapi:ProcessorResponseCode />
            <ipgapi:ProcessorApprovalCode />
            <ipgapi:ProcessorReceiptNumber />
            <ipgapi:ProcessorTraceNumber />
      </ipgapi:IPGApiOrderResponse>
</detail>
```

Erklärung:

Wurde eine korrekte Transaktion gesendet, kann ein Grund für diese Ablehnung sein, dass die Ihrer Store ID zugeordneten Terminal IDs für das Transaktionsvolumen nicht ausreichen. Bitte kontaktieren Sie Ihren Vertriebsansprechpartner um weitere Terminal IDs zur Lastverteilung zu bestellen.

Fehleranalyse - Login-Fehler bei der Verwendung von cURL

*** About to connect() to test.ipg-online.com port 443 (#0)**

*** Trying 217.73.32.55... connected**

*** Connected to test.ipg-online.com (217.73.32.55) port 443 (#0)**

*** unable to set private key file: 'C:\API\config\WS120666668._.1.key' type PEM**

*** Closing connection #0**

curl: (58) unable to set private key file: 'C:\API\config\WS120666668._.1.key' type PEM

Erklärung:

Keystore und Passwort passen nicht. Bitte überprüfen Sie Keystore und Passwort und ob Sie das **WS**<storeId>._.1.pem file verwendet haben. Ändern Sie die Dateiendung auf .cer, können Sie das Zertifikat mit einem Doppelklick öffnen. Bitte machen Sie diese Änderung nach der Überprüfung wieder rückgängig.

*** SSL certificate problem, verify that the CA cert is OK. Details:**

error:14090086:SSL routines:SSL3_GET_SERVER_CERTIFICATE:certificate verify failed

*** Closing connection #0**

curl: (60) SSL certificate problem, verify that the CA cert is OK. Details:

error:14090086:SSL routines:SSL3_GET_SERVER_CERTIFICATE:certificate verify failed

More details here: http://curl.haxx.se/docs/sslcerts.html

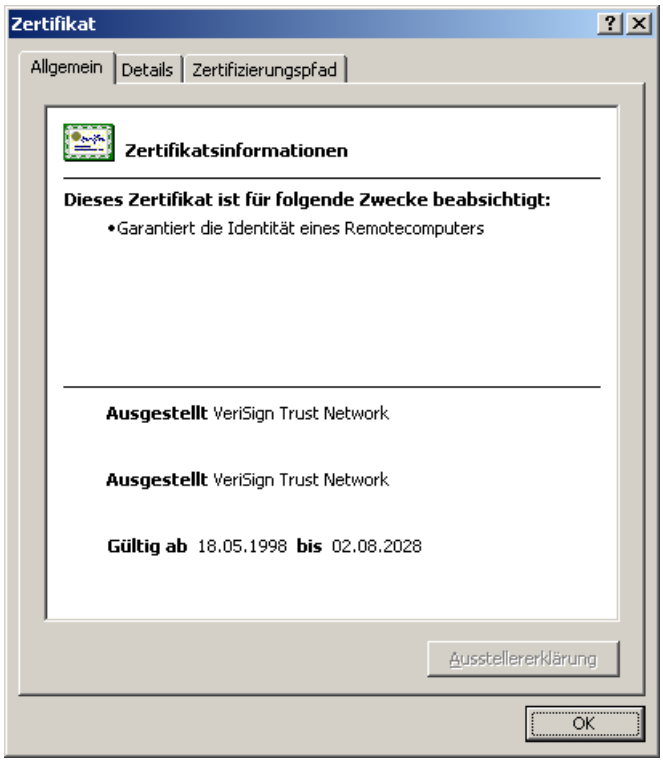
curl performs SSL certificate verification by default, using a "bundle" of Certificate Authority (CA) public keys (CA certs). The default bundle is named curl-ca-bundle.crt; you can specify an alternate file using the --cacert option.

If this HTTPS server uses a certificate signed by a CA represented in the bundle, the certificate verification probably failed due to a problem with the certificate (it might be expired, or the name might not match the domain name in the URL).

If you'd like to turn off curl's verification of the certificate, use the -k (or --insecure) option

Erklärung:

Das Truststore Zertifikat ist falsch. Bitte überprüfen Sie das Zertifikat auf den Issuer VeriSign:




```
<html>
	<head>
		<title>Apache Tomcat/5.5.20 - Error report</title>
		<style>
			<!--
H1 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:22px;}
H2 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:16px;}
H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:14px;}
BODY {font-family:Tahoma,Arial,sans-serif;color:black;background-color:white;}
B {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;}
P {font-family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;}
A {color : black;}
A.name {color : black;}
HR {color : #525D76;}
			-->
		</style>
	</head>
	<body>
		<h1>HTTP Status 401 - </h1>
		<HR size="1" noshade="noshade">
```

```
<p><b>type</b> Status report</p><p><b>message</b>
<u></u></p><p><b>description</b>
<u>This request requires HTTP authentication ().</u></p>
<HR size="1" noshade="noshade">
<h3>Apache Tomcat/5.5.20</h3>
</body>
</html>
```

Erklärung:

Das Zertifikat ist in Ordnung, jedoch sind User oder Passwort falsch.

Fehleranalyse - Login-Fehler bei der Verwendung des Java Client

java.io.IOException: Keystore was tampered with, or password was incorrect

Erklärung:

Keystore oder Truststore ist falsch. Sie können das Passwort mit dem Keytool, das Teil des JDK (bin directory) ist überprüfen:

c:\Programme\Java\jdk1.6.0_07\bin\keytool.exe -list -v -keystore <your keystore or truststore> -storepass <your keystore or truststore password>

javax.net.ssl.SSLHandshakeException: sun.security.validator.ValidatorException: No trusted certificate found

Erklärung:

Ihr Truststore ist falsch. Sie können das Truststore mit dem Keytool, das Teil des JDK ist überprüfen:

`c:\Programme\Java\jdk1.6.0_07\bin\keytool.exe -list -v -keystore` <Ihr Truststore> `-storepass` <Ihr Truststore Passwort>

Dabei muss sich der Herausgeber Verisign in der Ausgabe befinden:

```
OU=VeriSign Trust Network, OU="(c) 1998 VeriSign, Inc. - For authorized use
only", OU=Class 3 Public Primary Certification Authority - G2, O="VeriSign,
Inc.", C=US
```

Das Beispiel auf der folgenden Seite zeigt einen Ausgabe abhängig vom JDK. Bitte prüfen Sie auch die MD5 und SHA1 Werte.

```
C:\>c:\Programme\Java\jdk1.6.0_16\bin\keytool.exe  -list -v -keystore
truststore.jks -storepass IPGAPI

Keystore-Typ: JKS
Keystore-Provider: SUN

Ihr Keystore enthält 3 Einträge.

Aliasname: verisign_cacert
Erstellungsdatum: 26.11.2009
Eintragstyp: trustedCertEntry

Eigner: OU=VeriSign Trust Network, OU="(c) 1998 VeriSign, Inc. - For authorized use
only", OU=Class 3 Public Primary Certification Authority - G2, O="VeriSign, Inc.",
C=US
Aussteller: OU=VeriSign Trust Network, OU="(c) 1998 VeriSign, Inc. - For authorized
use only", OU=Class 3 Public Primary Certification Authority - G2, O="VeriSign,
Inc.", C=US
Seriennummer: 7dd9fe07cfa81eb7107967fba78934c6
Gültig von: Mon May 18 02:00:00 CEST 1998 bis: Wed Aug 02 01:59:59 CEST 2028
Digitaler Fingerabdruck des Zertifikats:
         MD5:  A2:33:9B:4C:74:78:73:D4:6C:E7:C1:F3:8D:CB:5C:E9
         SHA1: 85:37:1C:A6:E5:50:14:3D:CE:28:03:47:1B:DE:3A:09:E8:F8:77:0F
         Unterschrift-Algorithmusname: SHA1withRSA
         Version: 1


*******************************************
*******************************************


Aliasname: cacert
Erstellungsdatum: 23.07.2008
Eintragstyp: trustedCertEntry

Eigner: OU=Equifax Secure Certificate Authority, O=Equifax, C=US
Aussteller: OU=Equifax Secure Certificate Authority, O=Equifax, C=US
Seriennummer: 35def4cf
Gültig von: Sat Aug 22 18:41:51 CEST 1998 bis: Wed Aug 22 18:41:51 CEST 2018
Digitaler Fingerabdruck des Zertifikats:
         MD5:  67:CB:9D:C0:13:24:8A:82:9B:B2:17:1E:D1:1B:EC:D4
         SHA1: D2:32:09:AD:23:D3:14:23:21:74:E4:0D:7F:9D:62:13:97:86:63:3A
         Unterschrift-Algorithmusname: SHA1withRSA
         Version: 3

Erweiterungen:

#1: ObjectId: 2.5.29.16 Criticality=false
PrivateKeyUsage: [
To: Wed Aug 22 18:41:51 CEST 2018]

#2: ObjectId: 2.5.29.15 Criticality=false
KeyUsage [
  Key_CertSign
  Crl_Sign
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 48 E6 68 F9 2B D2 B2 95   D7 47 D8 23 20 10 4F 33  H.h.+....G.# .O3
0010: 98 90 9F D4                                        ....
]
]

#4: ObjectId: 1.2.840.113533.7.65.0 Criticality=false

#5: ObjectId: 2.5.29.31 Criticality=false
```

```
CRLDistributionPoints [
  [DistributionPoint:
     [CN=CRL1, OU=Equifax Secure Certificate Authority, O=Equifax, C=US]
]]

#6: ObjectId: 2.5.29.19 Criticality=false
BasicConstraints:[
  CA:true
  PathLen:2147483647
]

#7: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
KeyIdentifier [
0000: 48 E6 68 F9 2B D2 B2 95   D7 47 D8 23 20 10 4F 33  H.h.+....G.# .O3
0010: 98 90 9F D4                                        ....
]

]



*******************************************
*******************************************


Aliasname: verisign_g2
Erstellungsdatum: 26.11.2009
Eintragstyp: trustedCertEntry

Eigner: CN=VeriSign Class 3 Secure Server CA - G2, OU=Terms of use at
https://www.verisign.com/rpa (c)09, OU=VeriSign Trust Network, O="VeriSign, Inc.",
C=US
Aussteller: OU=VeriSign Trust Network, OU="(c) 1998 VeriSign, Inc. - For authorized
use only", OU=Class 3 Public Primary Certification Authority - G2, O="VeriSign,
Inc.", C=US
Seriennummer: 6e4ffab3c5e669c4d167c992abe858c4
Gültig von: Wed Mar 25 01:00:00 CET 2009 bis: Mon Mar 25 00:59:59 CET 2019
Digitaler Fingerabdruck des Zertifikats:
         MD5:  AE:0F:D7:09:45:EA:3C:10:60:B6:17:BC:8E:09:07:69
         SHA1: 62:F3:C8:97:71:DA:4C:E0:1A:91:FC:13:E0:2B:60:57:B4:54:7A:1D
         Unterschrift-Algorithmusname: SHA1withRSA
         Version: 3

Erweiterungen:

#1: ObjectId: 2.5.29.15 Criticality=true
KeyUsage [
  Key_CertSign
  Crl_Sign
]

#2: ObjectId: 2.5.29.19 Criticality=true
BasicConstraints:[
  CA:true
  PathLen:0
]

#3: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: A5 EF 0B 11 CE C0 41 03   A3 4A 65 90 48 B2 1C E0  ......A..Je.H...
0010: 57 2D 7D 47                                        W-.G
]
]

#4: ObjectId: 1.3.6.1.5.5.7.1.1 Criticality=false
AuthorityInfoAccess [
```

```
  [accessMethod: 1.3.6.1.5.5.7.48.1
   accessLocation: URIName: http://ocsp.verisign.com]
]

#5: ObjectId: 1.3.6.1.5.5.7.1.12 Criticality=false

#6: ObjectId: 2.5.29.31 Criticality=false
CRLDistributionPoints [
  [DistributionPoint:
     [URIName: http://crl.verisign.com/pca3-g2.crl]
]]

#7: ObjectId: 2.5.29.32 Criticality=false
CertificatePolicies [
  [CertificatePolicyId: [2.16.840.1.113733.1.7.23.3]
[PolicyQualifierInfo: [
  qualifierID: 1.3.6.1.5.5.7.2.1
  qualifier: 0000: 16 1C 68 74 74 70 73 3A   2F 2F 77 77 77 2E 76 65
..https://www.ve
0010: 72 69 73 69 67 6E 2E 63   6F 6D 2F 63 70 73        risign.com/cps

], PolicyQualifierInfo: [
  qualifierID: 1.3.6.1.5.5.7.2.2
  qualifier: 0000: 30 1E 1A 1C 68 74 74 70   73 3A 2F 2F 77 77 77 2E
0...https://www.
0010: 76 65 72 69 73 69 67 6E   2E 63 6F 6D 2F 72 70 61  verisign.com/rpa

]]  ]
]

#8: ObjectId: 2.5.29.35 Criticality=false
AuthorityKeyIdentifier [
[OU=VeriSign Trust Network, OU="(c) 1998 VeriSign, Inc. - For authorized use only",
OU=Class 3 Public Primary Certification Authority - G2, O="VeriSign, Inc
.", C=US]
SerialNumber: [    7dd9fe07 cfa81eb7 107967fb a78934c6]
]

#9: ObjectId: 2.5.29.17 Criticality=false
SubjectAlternativeName [
  CN=Class3CA2048-1-52
]



*****************************************
*****************************************
```

```
<html>
      <head>
            <title>Apache Tomcat/5.5.20 - Error report</title>
            <style><!--H1 {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;font-size:22px;} H2 {font-
family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-
size:16px;} H3 {font-family:Tahoma,Arial,sans-serif;color:white;background-
color:#525D76;font-size:14px;} BODY {font-family:Tahoma,Arial,sans-
serif;color:black;background-color:white;} B {font-family:Tahoma,Arial,sans-
serif;color:white;background-color:#525D76;} P {font-family:Tahoma,Arial,sans-
serif;background:white;color:black;font-size:12px;}A {color : black;}A.name {color
: black;}HR {color : #525D76;}--></style>
      </head>
      <body>
            <h1>HTTP Status 401 -</h1>
            <HR size="1" noshade="noshade">
```

```
            <p>
                  <b>type</b>
                  Status report
            </p>
            <p>
                  <b>message</b>
                  <u></u>
            </p>
            <p>
                  <b>description</b>
                  <u>This request requires HTTP authentication ().</u>
            </p>
            <HR size="1" noshade="noshade">
            <h3>Apache Tomcat/5.5.20</h3>
      </body>
</html>
```

Erklärung:

User ID oder Passwort ist falsch.